

0004

0000

0000
00000000
00000004
00040000
0000

0000

0000
00000000
0000

0000

0004
00040000
0000

314

485
7551

7FFI

7FFI

7FFI
7FFI

7FFI

7FF0
7FF17FPI
7FEI

7FFI

7F F1
3F F17F F1
7F F1

7FFC

7FF0
7FF1

7EFFI

7FFI

7FF0
7EE0

7FFI

7FF0

7FF1
7EE1

7FF0

7F F0
7F F07F F1
7F F1

7FFI

7FF
3FF

7F F1
7F F1

7FF

7FF

1

[illegible]

```
1 0001 0 MODULE ACCOUNTNG(XTITLE 'Accounting manager'
2 0002 0 IDENT = 'V04-000'
3 0003 0 ) =
4 0004 1 BEGIN
5 0005 1
6 0006 1
7 0007 1 *****
8 0008 1 *
9 0009 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
10 0010 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
11 0011 1 * ALL RIGHTS RESERVED.
12 0012 1 *
13 0013 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
14 0014 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
15 0015 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
16 0016 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
17 0017 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
18 0018 1 * TRANSFERRED.
19 0019 1 *
20 0020 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
21 0021 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
22 0022 1 * CORPORATION.
23 0023 1 *
24 0024 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
25 0025 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
26 0026 1 *
27 0027 1 *
28 0028 1 *****
29 0029 1
30 0030 1
31 0031 1 ++
32 0032 1 FACILITY:
33 0033 1 Job controller.
34 0034 1
35 0035 1 ABSTRACT:
36 0036 1 This module contains the accounting management routines.
37 0037 1
38 0038 1 ENVIRONMENT:
39 0039 1 VAX/VMS user and kernel mode.
40 0040 1 --
41 0041 1
42 0042 1 AUTHOR: M. Jack, CREATION DATE: 16-Feb-1982
43 0043 1
44 0044 1 MODIFIED BY:
45 0045 1
46 0046 1 V03-005 MHB0140 Mark Bramhall, 20-Apr-1984
47 0047 1 Change account name handling, especially the determination
48 0048 1 logic for generating SYSINIT and LOGFAIL records.
49 0049 1
50 0050 1 V03-004 MLJ0114 Martin L. Jack, 23-Jun-1983 4:48
51 0051 1 Changes for job controller baselevel.
52 0052 1
53 0053 1 V03-003 MLJ0113 Martin L. Jack, 26-May-1983 21:05
54 0054 1 Changes for job controller baselevel.
55 0055 1
56 0056 1 V03-002 MLJ0112 Martin L. Jack, 29-Apr-1983 2:48
57 0057 1 Changes for job controller baselevel.
```

ACCOUNTING
V04-000

Accounting manager

B 16
13-Sep-1984 23:46:25
14-Sep-1984 12:36:55

VAX-11 Bliss-32 V4.0-742
[JOBCTL.SRC]ACCOUNTING.B32;1

Page 2
(1)

.. 58
.. 59
.. 60
.. 61
.. 62

0058 1 |
0059 1 |
0060 1 |
0061 1 |
0062 1 |**

V03-001 MLJ0109 Martin L. Jack, 14-Apr-1983 12:45
Changes for job controller baselevel.

```
64 0063 1 REQUIRE 'SRC$:JOBCTLDEF';
65 1104 1
66 1105 1
67 1106 1 LITERAL
68 1107 1 ACCT_ACM_REG= 6,
69 1108 1 ACCT_ACR_REG= 7,
70 1109 1 ACCT_APK_REG= 8,
71 1110 1 ACCT_SJH_REG= 9,
72 1111 1 ACCT_SMQ_REG= 11;
73 1112 1
74 1113 1
75 1114 1 LINKAGE
76 1115 1 L_WRITE_ACCOUNTING_FILE= CALL: GLOBAL(
77 1116 1 ACR = ACCT_ACR_REG),
78 1117 1
79 1118 1 L_IDENT_PACKET= CALL: GLOBAL(
80 1119 1 ACM = ACCT_ACM_REG,
81 1120 1 ACR = ACCT_ACR_REG,
82 1121 1 SJH = ACCT_SJH_REG,
83 1122 1 SMQ = ACCT_SMQ_REG),
84 1123 1
85 1124 1 L_RESOURCE_PACKET= CALL: GLOBAL(
86 1125 1 ACM = ACCT_ACM_REG,
87 1126 1 ACR = ACCT_ACR_REG);
88 1127 1
89 1128 1
90 1129 1 FORWARD ROUTINE
91 1130 1 WRITE_ACCOUNTING_FILE: L_WRITE_ACCOUNTING_FILE NOVALUE,
92 1131 1 OPEN_ACCOUNTING_FILE: NOVALUE,
93 1132 1 CLOSE_ACCOUNTING_FILE: NOVALUE,
94 1133 1 IDENT_PACKET: L_IDENT_PACKET NOVALUE,
95 1134 1 RESOURCE_PACKET: L_RESOURCE_PACKET NOVALUE,
96 1135 1 WRITE_USER_ACCOUNTING_RECORD: NOVALUE,
97 1136 1 WRITE_FILE_LINK_RECORD: NOVALUE,
98 1137 1 WRITE_ACCOUNTING_RECORD: NOVALUE,
99 1138 1 WRITE_PRINT_RECORD: NOVALUE,
100 1139 1 WRITE_PROCESS_RECORD: NOVALUE,
101 1140 1 PROCESS_ACCOUNTING: NOVALUE;
102 1141 1
103 1142 1
104 1143 1 EXTERNAL ROUTINE
105 1144 1 FIND_PROCESS_DATA: L_OUTPUT_3,
106 1145 1 LOCK_QUEUE_FILE: NOVALUE,
107 1146 1 READ_RECORD,
108 1147 1 SIGNAL_FILE_ERROR: NOVALUE,
109 1148 1 UNLOCK_QUEUE_FILE: NOVALUE;
110 1149 1
111 1150 1
112 1151 1 EXTERNAL
113 1152 1 EXESGL_ACMFLAGS: BBLOCK ADDRESSING_MODE(GENERAL);
114 1153 1
115 1154 1
116 1155 1 BUILTIN
117 1156 1 LOCC,
118 1157 1 MOVCL,
119 1158 1 SKPC,
120 1159 1 TESTBITCC,
```

ACCOUNTNG
V04-000

Accounting manager

; 121

1160 1

TESTBITSC;

D 16
15-Sep-1984 23:46:25
14-Sep-1984 12:36:55

VAX-11 Bliss-32 V4.0-742
[JOBCTL.SRC]ACCOUNTNG.B32;1

Page 4
(2)

```
123 1161 1 ROUTINE WRITE_ACCOUNTING_FILE: L_WRITE_ACCOUNTING_FILE NOVALUE=
124 1162 1
125 1163 1 ++
126 1164 1
127 1165 1 FUNCTIONAL DESCRIPTION:
128 1166 1 This routine writes the accounting file.
129 1167 1
130 1168 1 INPUT PARAMETERS:
131 1169 1 NONE
132 1170 1
133 1171 1 IMPLICIT INPUTS:
134 1172 1 ACR - Pointer to accounting record.
135 1173 1
136 1174 1 OUTPUT PARAMETERS:
137 1175 1 NONE
138 1176 1
139 1177 1 IMPLICIT OUTPUTS:
140 1178 1 NONE
141 1179 1
142 1180 1 ROUTINE VALUE:
143 1181 1 NONE
144 1182 1
145 1183 1 SIDE EFFECTS:
146 1184 1 Accounting record written.
147 1185 1
148 1186 1 --
149 1187 1
150 1188 2 BEGIN
151 1189 2 EXTERNAL REGISTER
152 1190 2 ACR = ACCT_ACR_REG: REF BBLOCK; ! Pointer to accounting record
153 1191 2
154 1192 2
155 1193 2 ! The following loop is executed up to MAXFILERR times or until the accounting
156 1194 2 ! record is successfully written, provided that an accounting file is open and
157 1195 2 ! the record type is selected by the accounting control flags.
158 1196 2
159 1197 2 DECR I FROM JBC$K_MAXFILERR TO 1 DO
160 1198 3 BEGIN
161 1199 3 LOCAL
162 1200 3 FAB: REF BBLOCK; ! Pointer to FAB
163 1201 3
164 1202 3
165 1203 3 ! Pick up a pointer to the current accounting FAB. If the file is closed,
166 1204 3 ! return.
167 1205 3
168 1206 3 FAB = .ACCOUNTING_FABS[0];
169 1207 3 IF .FAB[FAB$W_IFI] NEQ 0
170 1208 3 THEN
171 1209 4 BEGIN
172 1210 4
173 1211 4 ! Evaluate the accounting control flags to determine whether the
174 1212 4 ! record type is currently selected.
175 1213 4
176 1214 4 IF
177 1215 5 BEGIN
178 1216 5 CASE .ACR[ACR$V_TYPE] FROM ACR$K_PRCDEL TO ACR$K_FILE_BL OF
179 1217 5 SET
```

```
180 1218 S
181 1219 S
182 1220 S
183 1221 S
184 1222 S
185 1223 S
186 1224 S
187 1225 S
188 1226 S
189 1227 S
190 1228 S
191 1229 S
192 1230 S
193 1231 S
194 1232 S
195 1233 S
196 1234 S
197 1235 S
198 1236 S
199 1237 S
200 1238 S
201 1239 S
202 1240 S
203 1241 S
204 1242 S
205 1243 S
206 1244 S
207 1245 S
208 1246 S
209 1247 S
210 1248 S
211 1249 S
212 1250 S
213 1251 S
214 1252 S
215 1253 S
216 1254 S
217 1255 S
218 1256 S
219 1257 S
220 1258 S
221 1259 S
222 1260 S
223 1261 S
224 1262 S
225 1263 S
226 1264 S
227 1265 S
228 1266 S
229 1267 S
230 1268 S
231 1269 S
232 1270 S
233 1271 S
234 1272 S
235 1273 S
236 1274 S
```

```
[OUTRANGE]:
  FALSE;
```

```
[ACR$K_PRCDEL, ACR$K_PRCPUR]:
  IF .EXE$GL_ACMFLAGS[ACM$V_PROCESS]
  THEN
    CASE .ACR[ACR$V_SUBTYPE] FROM ACR$K_INTERACTIVE TO ACR$K_NETWORK OF
      SET
      [OUTRANGE]: FALSE;
      [ACR$K_INTERACTIVE]: .EXE$GL_ACMFLAGS[ACM$V_INTERACTIVE];
      [ACR$K_SUBPROCESS]: .EXE$GL_ACMFLAGS[ACM$V_SUBPROCESS];
      [ACR$K_DETACHED]: .EXE$GL_ACMFLAGS[ACM$V_DETACHED];
      [ACR$K_BATCH]: .EXE$GL_ACMFLAGS[ACM$V_BATCH];
      [ACR$K_NETWORK]: .EXE$GL_ACMFLAGS[ACM$V_NETWORK];
    TES
  ELSE
    FALSE;
```

```
[ACR$K_IMGDEL, ACR$K_IMGPUR]:
  CASE .ACR[ACR$V_SUBTYPE] FROM ACR$K_INTERACTIVE TO ACR$K_NETWORK OF
    SET
    [OUTRANGE]: FALSE;
    [ACR$K_INTERACTIVE]: .EXE$GL_ACMFLAGS[ACM$V_INTERACTIVE];
    [ACR$K_SUBPROCESS]: .EXE$GL_ACMFLAGS[ACM$V_SUBPROCESS];
    [ACR$K_DETACHED]: .EXE$GL_ACMFLAGS[ACM$V_DETACHED];
    [ACR$K_BATCH]: .EXE$GL_ACMFLAGS[ACM$V_BATCH];
    [ACR$K_NETWORK]: .EXE$GL_ACMFLAGS[ACM$V_NETWORK];
  TES;
```

```
[ACR$K_SYSINIT, ACR$K_SETTIME, ACR$K_ENABLE, ACR$K_DISABLE,
  ACR$K_ALTACM, ACR$K_FILE_FL, ACR$K_FILE_BL]:
  TRUE;
```

```
[ACR$K_LOGFAIL]:
  .EXE$GL_ACMFLAGS[ACM$V_LOGFAIL];
```

```
[ACR$K_PRINT]:
  .EXE$GL_ACMFLAGS[ACM$V_PRINT];
```

```
[ACR$K_USER]:
  .EXE$GL_ACMFLAGS[ACM$V_USER_DATA];
```

```
TES
END
THEN BEGIN
  LABEL
  WRITE_RECORD;
```

```
237      1275 5      LOCAL
238      1276 5      RAB:          REF BBLOCK;      ! Pointer to RAB
239      1277 5
240      1278 5
241      1279 5  WRITE_RECORD:
242      1280 6      BEGIN
243      1281 6
244      1282 6      ! Pick up a pointer to the RAB.
245      1283 6
246      1284 6      RAB = .ACCOUNTING_RABS[0];
247      1285 6
248      1286 6
249      1287 6      ! If there is a previous asynchronous operation in progress, wait
250      1288 6      ! for its completion.
251      1289 6
252      1290 6  IF TESTBITSC(RAB[RAB$V_ASY])
253      1291 6  THEN
254      1292 7      BEGIN
255      1293 7      IF NOT $WAIT(RAB=.RAB) THEN LEAVE WRITE_RECORD;
256      1294 6      END;
257      1295 6
258      1296 6
259      1297 6      ! Initialize the record descriptor and write this record.
260      1298 6
261      1299 6      RAB[RAB$W_RSZ] = .ACR[ACR$W_LENGTH];
262      1300 6      RAB[RAB$L_RBF] = .ACR;
263      1301 6      IF NOT $POT(RAB=.RAB) THEN LEAVE WRITE_RECORD;
264      1302 6
265      1303 6
266      1304 6      ! Unless this is an image accounting record, start an asynchronous
267      1305 6      ! $FLUSH to write this record to disk.
268      1306 6
269      1307 7  IF NOT ONEOF_(.ACR[ACR$V_TYPE], BMSK_(ACR$K_IMGDEL, ACR$K_IMGPUR))
270      1308 6  THEN
271      1309 7      BEGIN
272      1310 7      RAB[RAB$V_ASY] = TRUE;
273      1311 7      IF NOT $F[USH(RAB=.RAB) THEN LEAVE WRITE_RECORD;
274      1312 6      END;
275      1313 6
276      1314 6
277      1315 6      ! Completed successfully -- return.
278      1316 6
279      1317 6  RETURN;
280      1318 5  END;      ! block WRITE_RECORD
281      1319 5
282      1320 5
283      1321 5      ! An error occurred writing the record. Report it.
284      1322 5
285      1323 5      SIGNAL_FILE_ERROR(JBC$_WRITEERR + STS$K_ERROR, .FAB, .RAB);
286      1324 5
287      1325 5
288      1326 5      ! Unless the error is 'device full', close this accounting file,
289      1327 5      ! try to open a new one, and then try to write the record again.
290      1328 5
291      1329 5  IF .RAB[RAB$L_STS] EQL RMS$_FUL
292      1330 5  THEN
293      1331 5      EXITLOOP
```

```
294      1332 5      ELSE
295      1333 5      OPEN_ACCOUNTING_FILE(TRUE);
296      1334 5      END
297      1335 4      ELSE
298      1336 4      RETURN;
299      1337 4      END
300      1338 3      ELSE
301      1339 3      RETURN;
302      1340 2      END;
303      1341 2
304      1342 2
305      1343 2      ! Writing has failed. Implicitly disable accounting.
306      1344 2
307      1345 2      SIGNAL(JBC$ ACCDISERR OR ST$K_INFO);
308      1346 2      CLOSE_ACCOUNTING_FILE();
309      1347 1      END;
```

```
.TITLE ACCOUNTNG Accounting manager
.IDENT \V04-000\
.PSECT COMMON,NOEXE, OVR,2
```

```
00000 DIAG_STORAGE BASE:
      .BLKB 0
00000 DIAG_TRACE:
      .BLKB 96
00060 DIAG_COUNT:
      .BLKB 96
000C0 DIAG_FLAGS:
      .BLKB 4
000C4 WORK_AREA:
      .BLKB 44
000F0 SNDJBC_COUNT:
      .BLKB 132
00174 GETQUI_COUNT:
      .BLKB 40
0019C SNDACC_COUNT:
      .BLKB 28
001B8 SNDSMB_COUNT:
      .BLKB 72
00200 DIAG_STORAGE_END:
      .BLKB 0
00200 FLAGS:
      .BLKB 4
00204 IMAGE_DUMP_STSFLG:
      .BLKB 4
00208 THIS_SYSID:
      .BLKB 6
0020E
      .BLKB 2
00210 CUR_TIME:
      .BLKB 8
00218 HOURLY_TIME:
      .BLKB 8
00220 HOURLY_PARAMS:
      .BLKB 20
00234 SYMBIONT_COUNT:
      .BLKB 4
```

0023B	QUEUE_REFERENCE_COUNT:	.BLKB	4
0023C	MBX_MESSAGE_COUNT:	.BLKB	4
00240	MBX:	.BLKB	4
00244	MBX_END:	.BLKB	4
00248	MEMORY_FREE_QUEUES:	.BLRB	40
00270	NONAST_WORK_QUEUE:	.BLRB	8
00278	BCB_FREE_LIST:	.BLKB	4
0027C	BCB_ACTIVE_LIST:	.BLKB	4
00280	GQL_FREE_LIST:	.BLKB	4
00284	GQL_ACTIVE_LIST:	.BLKB	4
00288	OPEN_GETQUI_LIST:	.BLRB	4
0028C	PROCESS_DATA_LIST:	.BLKB	4
00290	SYMBIONT_CONTROL:	.BLKB	4
00294	SPARE_AREA:	.BLKB	12
002A0	REMOTE_REQUEST_LKSB:	.BLKB	8
002A8	QUEUE_FILE_LKSB:	.BLKB	8
002B0	QUEUE_LOCK_LKSB:	.BLKB	8
002B8	RSP:	.BLKB	8
002C0	JBC_PRIORITY:	.BLKB	4
002C4	JBC_PRIVILEGES:	.BLKB	8
002CC	JBC_QUOTAS:	.BLKB	66
0030E		.BLKB	2
00310	JBC_UIC:	.BLKB	4
00314	QUEUE_FAB:	.BLKB	80
00364	QUEUE_RAB:	.BLKB	68
003A8	QUEUE_NAM:	.BLKB	96
00408	QUEUE_XAB:	.BLKB	88
00460	QUEUE_RSA:	.BLKB	255
0055F		.BLKB	1
00560	QUEUE_ALQ:	.BLKB	4
00564	QUEUE_MBF:	.BLKB	1
00565		.BLKB	3

J 16
15-Sep-1984 23:46:25
14-Sep-1984 12:36:55VAX-11 Bliss-32 V4.0-742
[JOBCTL.SRC]ACCOUNTNG.B32;1

00568 ACCOUNTING_FABS:
 .BCKB 8
00570 ACCOUNTING_RABS:
 .BCKB 8
00578 ACCOUNT_FAB_A:
 .BLRB 80
005C8 ACCOUNT_RAB_A:
 .BLRB 68
0060C ACCOUNT_NAM_A:
 .BLRB 96
0066C ACCOUNT_RSA_A:
 .BLRB 255
0076B .BLKB 1
0076C ACCOUNT_FAB_B:
 .BLRB 80
007BC ACCOUNT_RAB_B:
 .BLRB 68
00800 ACCOUNT_NAM_B:
 .BLRB 96
00860 ACCOUNT_RSA_B:
 .BLRB 255
0095F .BLKB 1
00960 DIAG_FAB:
 .BLKB 80
00980 DIAG_RAB:
 .BLKB 68
009F4 MBX_CHAN:
 .BLKB 4
009F8 MBX_IOSB:
 .BLKB 8
00A00 MBX_BUFFER:
 .BLKB 1024
00E00 VALUE_STORAGE_BASE:
 .BLKB 0
00E00 ITEM_PRESENT:
 .BLKB 32
00E20 VALUE_GETQUI_BASE:
 .BLKB 0
00E20 VALUE_ACCOUNTING_MESSAGE:
 .BLKB 8
00E26 VALUE_ACCOUNTING_TYPES:
 .BLKB 4
00E2A VALUE_AFTER_TIME:
 .BLRB 8
00E32 VALUE_ALIGNMENT_PAGES:
 .BLKB 1
00E33 VALUE_BASE_PRIORITY:
 .BCKB 1
00E34 VALUE_BATCH_INPUT:
 .BLRB 6
00E3A VALUE_BATCH_OUTPUT:
 .BLRB 10
00E44 VALUE_BUFFER_COUNT:
 .BLKB 1
00E45 VALUE_CHARACTERISTIC_NAME:
 .BLKB 6
00E4B VALUE_CHARACTERISTIC_NUMBER:

```

                                .BLKB 1
00E4C VALUE_CHARACTERISTICS:
                                .BLKB 16
00E5C VALUE_CHECKPOINT_DATA:
                                .BLKB 8
00E62 VALUE_CLI:
                                .BLKB 6
00E68 VALUE_CPU_DEFAULT:
                                .BLKB 4
00E6C VALUE_CPU_LIMIT:
                                .BLKB 4
00E70 VALUE_DESTINATION_QUEUE:
                                .BLKB 8
00E78 VALUE_DEVICE_NAME:
                                .BLKB 6
00E7E VALUE_ENTRY_NUMBER:
                                .BLKB 4
00E82 VALUE_ENTRY_NUMBER_OUTPUT:
                                .BLKB 10
00E8C VALUE_EXTEND_QUANTITY:
                                .BLKB 2
00E8E VALUE_FILE_COPIES:
                                .BLKB 1
00E8F VALUE_FILE_IDENTIFICATION:
                                .BLKB 36
00EB3 VALUE_FILE_SETUP_MODULES:
                                .BLKB 8
00EB9 VALUE_FILE_SPECIFICATION:
                                .BLKB 6
00EBF VALUE_FIRST_PAGE:
                                .BLKB 4
00EC3 VALUE_FORM_DESCRIPTION:
                                .BLKB 6
00EC9 VALUE_FORM_LENGTH:
                                .BLKB 1
00ECA VALUE_FORM_MARGIN_BOTTOM:
                                .BLKB 1
00ECB VALUE_FORM_MARGIN_LEFT:
                                .BLKB 2
00ECD VALUE_FORM_MARGIN_RIGHT:
                                .BLKB 2
00ECF VALUE_FORM_MARGIN_TOP:
                                .BLKB 1
00ED0 VALUE_FORM_NAME:
                                .BLKB 6
00ED6 VALUE_FORM_NUMBER:
                                .BLKB 4
00EDA VALUE_FORM:
                                .BLKB 8
00EE2 VALUE_FORM_SETUP_MODULES:
                                .BLKB 8
00EE8 VALUE_FORM_STOCK:
                                .BLKB 6
00EEE VALUE_FORM_WIDTH:
                                .BLKB 2
00EFO VALUE_GENERIC_TARGET:
                                .BLKB 996

```

15-Sep-1984 23:46:25
14-Sep-1984 12:36:55

VAX-11 BLISS-32 V4.0-742
[JOBCTL.SRC]ACCOUNTING.B32;1

Page 12
(3)

```

012D4 VALUE_JOB_COPIES:
      .BLKB 1
012D5 VALUE_JOB_LIMIT:
      .BLKB 1
012D6 VALUE_JOB_NAME:
      .BLKB 6
012DC VALUE_JOB_RESET_MODULES:
      .BLKB 6
012E2 VALUE_JOB_SIZE_MAXIMUM:
      .BLKB 4
012E6 VALUE_JOB_SIZE_MINIMUM:
      .BLKB 4
012EA VALUE_JOB_STATUS_OUTPUT:
      .BLKB 10
012F4 VALUE_LAST_PAGE:
      .BLKB 4
012F8 VALUE_LIBRARY_SPECIFICATION:
      .BLKB 6
012FE VALUE_LOG_QUEUE:
      .BLKB 8
01306 VALUE_LOG_SPECIFICATION:
      .BLKB 6
0130C VALUE_NOTE:
      .BLKB 6
01312 VALUE_OPERATOR_REQUEST:
      .BLKB 6
01318 VALUE_OWNER_UID:
      .BLKB 4
0131C VALUE_PAGE_SETUP_MODULES:
      .BLKB 6
01322 VALUE_PARAMETER_1:
      .BLKB 6
01328 VALUE_PARAMETER_2:
      .BLKB 6
0132E VALUE_PARAMETER_3:
      .BLKB 6
01334 VALUE_PARAMETER_4:
      .BLKB 6
0133A VALUE_PARAMETER_5:
      .BLKB 6
01340 VALUE_PARAMETER_6:
      .BLKB 6
01346 VALUE_PARAMETER_7:
      .BLKB 6
0134C VALUE_PARAMETER_8:
      .BLKB 6
01352 VALUE_PRIORITY:
      .BLKB 1
01353 VALUE_PROCESSOR:
      .BLKB 6
01359 VALUE_PROTECTION:
      .BLKB 4
0135D VALUE_QUEUE:
      .BLKB 6
01363 VALUE_QUEUE_FILE_SPECIFICATION:
      .BLKB 6
01369 VALUE_RELATIVE_PAGE:

```

.BLKB 4
0136D VALUE_RESERVED_INPUT_1:
.BLKB 1
0136E VALUE_RESERVED_INPUT_2:
.BLKB 2
01370 VALUE_RESERVED_INPUT_3:
.BLKB 4
01374 VALUE_RESERVED_INPUT_4:
.BLKB 6
0137A VALUE_RESERVED_OUTPUT_1:
.BLKB 10
01384 VALUE_RESERVED_OUTPUT_2:
.BLKB 10
0138E VALUE_SEARCH_STRING:
.BLKB 6
01394 VALUE_SC\$NODE_NAME:
.BLKB 6
0139A VALUE_WSDEFAULT:
.BLKB 2
0139C VALUE_W\$EXTENT:
.BLKB 2
0139E VALUE_W\$QUOTA:
.BLKB 2
013A0 VALUE_STORAGE_END:
.BLKB 0

JBC\$_CLOSEOUT= 266328
JBC\$_NOCMKRNL= 272388
JBC\$_NOOPER= 272532
JBC\$_NOSYSNAM= 272404
JBC\$_OPENIN= 266392
JBC\$_OPENOUT= 266400
JBC\$_READERR= 266416
JBC\$_WRITEERR= 266448
.EXTRN FIND_PROCESS_DATA
.EXTRN LOCK_QUEUE_FILE
.EXTRN READ_RECORD, SIGNAL_FILE_ERROR
.EXTRN UNLOCK_QUEUE_FILE
.EXTRN EXE\$GL_ACMFLAGS
.EXTRN SYSSWAIT, SYSSPUT
.EXTRN SYSSFLUSH
.PSECT CODE, NOWRT, 2

003C 00000 WRITE_ACCOUNTING_FILE:

55	00000000G	00	9E	00002	.WORD	Save R2, R3, R4, R5	1161
54		02	D0	00009	MOVAB	EXE\$GL_ACMFLAGS, R5	
53	00000000'	EF	D0	0000C	1\$: MOVL	#2, I	1197
	02	A3	B5	00013	MOVL	ACCOUNTING_FABS, FAB	1206
		01	12	00016	TS1W	2(FAB)	1207
			04	00018	BNEQ	2\$	
			01	EF	RET		
50	67	07	01	EF	2\$: EXTZV	#1, #7, (ACR), R0	1216
	0D	01	50	CF	CASEL	R0, #1, #13	
0036	0036	001D	001D	00022	3\$: .WORD	4\$-3\$,-	
0069	0064	0074	0074	0002A		4\$-3\$,-	
0074	0074	0074	006E	00032		7\$-3\$,-	

0074

0074

0003A

78-38,-
178-38,-
178-38,-
148-38,-
158-38,-
168-38,-
178-38,-
178-38,-
178-38,-
178-38,-
178-38,-

		01		65	04 0003E	RET			
					E8 0003F 48:	BLBS	EXESGL_ACMFLAGS, 58		1225
					04 00042	RET			
51	01	A7	04	00	EF 00043 58:	EXTZV	#0, #4, 1(ACR), R1		1227
		04	01	51	CF 00049	CASEL	R1, #1, #4		
002F		002A	0025	0020	0004D 68:	.WORD	98-68,-		
				0034	00055		108-68,-		
							118-68,-		
							128-68,-		
							138-68		
					04 00057	RET			
51	01	A7	04	00	EF 00058 78:	EXTZV	#0, #4, 1(ACR), R1		1241
		04	01	51	CF 0005E	CASEL	R1, #1, #4		
001A		0015	0010	000B	00062 88:	.WORD	98-88,-		
				001F	0006A		108-88,-		
							118-88,-		
							128-88,-		
							138-88		
					04 0006C	RET			
		25	65	02	E0 0006D 98:	BBS	#2, EXESGL_ACMFLAGS, 178		1244
					04 00071	RET			
		20	65	04	E0 00072 108:	BBS	#4, EXESGL_ACMFLAGS, 178		1245
					04 00076	RET			
		18	65	05	E0 00077 118:	BBS	#5, EXESGL_ACMFLAGS, 178		1246
					04 0007B	RET			
		16	65	06	E0 0007C 128:	BBS	#6, EXESGL_ACMFLAGS, 178		1247
					04 00080	RET			
				65	95 00081 138:	TSTB	EXESGL_ACMFLAGS		1248
				11	19 00083	BLSS	178		
					04 00085	RET			
		0C	65	03	E0 00086 148:	BBS	#3, EXESGL_ACMFLAGS, 178		1258
					04 0008A	RET			
			07	01	A5 E8 00C8B 158:	BLBS	EXESGL_ACMFLAGS+1, 178		1262
					04 0008F	RET			
		01	01	A5	E0 00090 168:	BBS	#1, EXESGL_ACMFLAGS+1, 178		1266
					04 00095	RET			
			52	00000000'	EF D0 00096 178:	MOVL	ACCOUNTING_RABS, RAB		1284
			A2	00	E5 0009D	BBCC	#0, 4(RAB), 188		1290
				52	DD 000A2	PUSHL	RAB		1293
		00000000G	00	01	FB 000A4	CALLS	#1, SYS\$WAIT		
			34	50	E9 000AB	BLBC	R0, 198		
		22	A2	02	A7 B0 000AE 188:	MOVW	2(ACR), 34(RAB)		1299
		28	A2	57	D0 000B3	MOVL	ACR, 40(RAB)		1300
				52	DD 000B7	PUSHL	RAB		1301
		00000000G	00	01	FB 000B9	CALLS	#1, SYS\$PUT		
			1F	50	E9 000C0	BLBC	R0, 198		

50	67	07	01	EF	000C3	EXTZV	#1, #7, (ACR), R0	1307
	50	8F	50	78	000C8	ASHL	R0, #402653184, R0	
			4C	19	000D0	BLSS	22\$	
	04	A2	01	88	000D2	BISB2	#1, 4(RAB)	1310
			52	DD	000D6	PUSHL	RAB	1311
	00000000G	00	01	FB	000D8	CALLS	#1, SYSSFLUSH	
		3C	50	E8	000DF	BLBS	R0, 22\$	
			52	DD	000E2	PUSHL	RAB	1323
			53	DD	000E4	PUSHL	FAB	
		000410D2	8F	DD	000E6	PUSHL	#266450	
	00000000G	EF	03	FB	000EC	CALLS	#3, SIGNAL_FILE_ERROR	
	00018544	8F	A2	D1	000F3	CMPL	8(RAB), #99652	1329
			0F	13	000FB	BEQL	21\$	
			01	DD	000FD	PUSHL	#1	1333
	0000V	CF	01	FB	000FF	CALLS	#1, OPEN_ACCOUNTING_FILE	
		02	54	F5	00104	SOBGTR	I, 20\$	1197
			03	11	00107	BRB	21\$	
			FF00	31	00109	BRW	1\$	
		00048403	8F	DD	0010C	PUSHL	#295939	1345
	00000000G	00	01	FB	00112	CALLS	#1, LIB\$SIGNAL	
	0000V	CF	00	FB	00119	CALLS	#0, CLOSE_ACCOUNTING_FILE	1346
			04	0011E	22\$:	RET		1347

; Routine Size: 287 bytes, Routine Base: CODE + 0000

```
311 1348 1 GLOBAL ROUTINE OPEN_ACCOUNTING_FILE(NEW): NOVALUE=
312 1349 1
313 1350 1 **
314 1351 1
315 1352 1 FUNCTIONAL DESCRIPTION:
316 1353 1 This routine opens an accounting file. If one is already open, it is
317 1354 1 closed and a new copy created.
318 1355 1
319 1356 1 INPUT PARAMETERS:
320 1357 1 NEW - True if a new file must be created.
321 1358 1
322 1359 1 IMPLICIT INPUTS:
323 1360 1 NONE
324 1361 1
325 1362 1 OUTPUT PARAMETERS:
326 1363 1 NONE
327 1364 1
328 1365 1 IMPLICIT OUTPUTS:
329 1366 1 NONE
330 1367 1
331 1368 1 ROUTINE VALUE:
332 1369 1 NONE
333 1370 1
334 1371 1 SIDE EFFECTS:
335 1372 1 Accounting file opened.
336 1373 1
337 1374 1 --
338 1375 1
339 1376 2 BEGIN
340 1377 2 LOCAL
341 1378 2 OLD_IFI, : Previous IFI value for current file
342 1379 2 OLD_FAB: REF BBLOCK, : Current FAB
343 1380 2 OLD_RAB: REF BBLOCK, : Current RAB
344 1381 2 NEW_FAB: REF BBLOCK, : Next FAB
345 1382 2 NEW_RAB: REF BBLOCK, : Next RAB
346 1383 2
347 1384 2
348 1385 2 : If this is the first call, initialize.
349 1386 2
350 1387 2 IF .ACCOUNTING_FABS[0] EQL 0
351 1388 2 THEN
352 1389 2 BEGIN
353 1390 2 ACCOUNTING_FABS[0] = ACCOUNT_FAB_A;
354 1391 2 ACCOUNTING_FABS[1] = ACCOUNT_FAB_B;
355 1392 2 ACCOUNTING_RABS[0] = ACCOUNT_RAB_A;
356 1393 2 ACCOUNTING_RABS[1] = ACCOUNT_RAB_B;
357 1394 2 $FAB_INIT(FAB=ACCOUNT_FAB_A,
358 1395 2 FAC=PUT,
359 1396 2 FNA=UPLIT BYTE('ACCOUNTNG'),
360 1397 2 FNS=XCHARCOUNT('ACCOUNTNG'),
361 1398 2 DNA=UPLIT BYTE('SYSSMANAGER:.DAT'),
362 1399 2 DNS=XCHARCOUNT('SYSSMANAGER:.DAT'),
363 1400 2 FOP=CIF,
364 1401 2 DEQ=25,
365 1402 2 ORG=SEQ,
366 1403 2 RFM=VAR,
367 1404 2 NAM=ACCOUNT_NAM_A,
```

```
368 1405 3 SHR=<GET,UPI>;
369 1406 3 SRAB_INIT(RAB=ACCOUNT_RAB_A,
370 1407 3 FAB=ACCOUNT_FAB_A,
371 1408 3 MBC=1,
372 1409 3 MBF=2,
373 1410 3 RAC=SEQ,
374 1411 3 ROP=<EOF,WBH>);
375 1412 3 $NAM_INIT(NAM=ACCOUNT_NAM_A,
376 1413 3 RSA=ACCOUNT_RSA_A,
377 1414 3 RSS=NAM$C_MAXRSS);
378 1415 3 CHSMOVE(FAB$C_BLN, ACCOUNT_FAB_A, ACCOUNT_FAB_B);
379 1416 3 CHSMOVE(RAB$C_BLN, ACCOUNT_RAB_A, ACCOUNT_RAB_B);
380 1417 3 CHSMOVE(NAM$C_BLN, ACCOUNT_NAM_A, ACCOUNT_NAM_B);
381 1418 3 ACCOUNT_FAB_B[FAB$L_NAM] = ACCOUNT_NAM_B;
382 1419 3 ACCOUNT_RAB_B[RAB$L_FAB] = ACCOUNT_FAB_B;
383 1420 3 ACCOUNT_NAM_B[NAM$L_RSA] = ACCOUNT_RSA_B;
384 1421 3 END;
385 1422 3
386 1423 3
387 1424 3 ! Pick up pointers to the current and new FAB and RAB.
388 1425 3
389 1426 3 OLD_FAB = .ACCOUNTING_FABS[0];
390 1427 3 NEW_FAB = .ACCOUNTING_FABS[1];
391 1428 3 OLD_RAB = .ACCOUNTING_RABS[0];
392 1429 3 NEW_RAB = .ACCOUNTING_RABS[1];
393 1430 3
394 1431 3
395 1432 3 ! If an accounting file is currently open, unconditionally create a new file.
396 1433 3 ! Otherwise, use the CIF option to connect to the end of the existing file.
397 1434 3 ! Set up ALQ for no initial allocation.
398 1435 3
399 1436 3 NEW_FAB[FAB$V_CIF] = TRUE;
400 1437 3 OLD_IFI = .OLD_FAB[FAB$W_IFI];
401 1438 3 IF .OLD_IFI NEQ 0 OR .NEW THEN NEW_FAB[FAB$V_CIF] = FALSE;
402 1439 3
403 1440 3
404 1441 3 ! Create or open the file. Accept an error that occurs during a create-if,
405 1442 3 ! and loop to create a new version.
406 1443 3
407 1444 3 WHILE TRUE DO
408 1445 3 BEGIN
409 1446 3 NEW_FAB[FAB$L_ALQ] = 0;
410 1447 3 IF NOT $CREATE(FAB=.NEW_FAB)
411 1448 3 THEN
412 1449 3 BEGIN
413 1450 3 IF TESTBITCC(NEW_FAB[FAB$V_CIF])
414 1451 3 THEN
415 1452 3 BEGIN
416 1453 3 SIGNAL_FILE_ERROR(JBC$_OPENOUT + STS$_ERROR, .NEW_FAB, .NEW_FAB);
417 1454 3 EXITLOOP;
418 1455 3 END;
419 1456 3 END
420 1457 3 ELSE
421 1458 3 BEGIN
422 1459 3 IF NOT $CONNECT(RAB=.NEW_RAB)
423 1460 3 THEN
424 1461 3 BEGIN
```

```

425 1462 S SIGNAL_FILE_ERROR(JBC$_OPENOUT + STS$_ERROR, .NEW_FAB, .NEW_RAB);
426 1463 S $CLOSE(FAB=.NEW_FAB);
427 1464 S NEW_FAB[FAB$_IF1] = 0;
428 1465 S END;
429 1466 S EXITLOOP;
430 1467 S END;
431 1468 S END;
432 1469 S
433 1470 S
434 1471 S ! If an accounting file was previously open and a new file has been created,
435 1472 S write the file forward link record. Omit this if forced creation of a new
436 1473 S file is requested, since the previous file sustained an error.
437 1474 S
438 1475 S IF (NOT .NEW_FAB[FAB$_CIF] OR .NEW_FAB[FAB$_STS] EQL RMS$_CREATED)
439 1476 S AND .OLD_IF1 NEQ 0
440 1477 S THEN
441 1478 S BEGIN
442 1479 S IF NOT .NEW THEN WRITE_FILE_LINK_RECORD(ACR$_FILE_FL, .NEW_FAB);
443 1480 S CLOSE_ACCOUNTING_FILE();
444 1481 S END;
445 1482 S
446 1483 S
447 1484 S ! Exchange FAB and RAB pointers.
448 1485 S
449 1486 S ACCOUNTING_FABS[0] = .NEW_FAB;
450 1487 S ACCOUNTING_FABS[1] = .OLD_FAB;
451 1488 S ACCOUNTING_RABS[0] = .NEW_RAB;
452 1489 S ACCOUNTING_RABS[1] = .OLD_RAB;
453 1490 S
454 1491 S
455 1492 S ! If an accounting file was previously open, write the file back link record.
456 1493 S
457 1494 S IF .OLD_IF1 NEQ 0 THEN WRITE_FILE_LINK_RECORD(ACR$_FILE_BL, .OLD_FAB);
458 1495 S END;

```

```

41 44 2E 3A 52 45 47 4E 54 4E 55 4F 43 43 41 0011F P.AAA: .ASCII \ACCOUNTNG\
53 00128 P.AAB: .ASCII \SYSSMANAGER:.DAT\
54 00137

```

```

$RMS_PTR= ACCOUNT_FAB_A
$RMS_PTR= ACCOUNT_RAB_A
$RMS_PTR= ACCOUNT_NAM_A
.EXTRN SYSS$CREATE, SYSS$CONNECT
.EXTRN SYSS$CLOSE

01FC 00000 .ENTRY OPEN_ACCOUNTING_FILE, Save R2,R3,R4,R5,R6,- : 1348
R7,R8
MOVAB SIGNAL_FILE_ERROR, R8
MOVAB ACCOUNT_FAB_A, R7
TSTL ACCOUNTING_FABS : 1387
BEQL 1$
BRW 2$
FO A7 67 9E 00018 1$: MOVAB ACCOUNT_FAB_A, ACCOUNTING_FABS : 1390
F4 A7 01F4 C7 9E 0001C MOVAB ACCOUNT_FAB_B, ACCOUNTING_FABS+4 : 1391
F8 A7 50 A7 9E 00022 MOVAB ACCOUNT_RAB_A, ACCOUNTING_RABS : 1392

```

0050	8F	00	FC	A7	0244	C7	9E	00027	MOVAB	ACCOUNT_RAB_B, ACCOUNTING_RABS+4	1393
				6E		00	2C	0002D	MOVCS	#0, (SPT), #0, #80, \$RMS_PTR	1405
				67	5003	67		00034			
			04	A7	02000000	8F	B0	00035	MOVW	#20483, \$RMS_PTR	
			14	A7	42010019	8F	D0	0003A	MOVL	#33554432, \$RMS_PTR+4	
					1D	8F	D0	00042	MOVL	#1107361817, \$RMS_PTR+20	
			1F	A7		A7	94	0004A	CLRB	\$RMS_PTR+29	
			28	A7	0094	02	90	0004D	MOVW	#2, \$RMS_PTR+31	
			2C	A7	8D	C7	9E	00051	MOVAB	ACCOUNT_NAM_A, \$RMS_PTR+40	
			30	A7	91	AF	9E	00057	MOVAB	P.AAA, \$RMS_PTR+44	
			34	A7	1009	AF	9E	0005C	MOVAB	P.AAB, \$RMS_PTR+48	
0044	8F	00		6E		8F	B0	00061	MOVW	#4105, \$RMS_PTR+52	
					50	00	2C	00067	MOVCS	#0, (SP), #0, #68, \$RMS_PTR	1411
					4401	A7		0006E			
			50	A7	0500	8F	B0	00070	MOVW	#17409, \$RMS_PTR	
			54	A7	6E	8F	3C	00076	MOVZWL	#1280, \$RMS_PTR+4	
					0102	A7	94	0007C	CLRB	\$RMS_PTR+30	
0060	8F	00	0086	C7		8F	B0	0007F	MOVW	#258, \$RMS_PTR+54	
			008C	C7		67	9E	00086	MOVAB	ACCOUNT_FAB_A, \$RMS_PTR+60	
				6E		00	2C	0008B	MOVCS	#0, (SPT), #0, #96, \$RMS_PTR	1414
					0094	C7		00092			
					6002	8F	B0	00095	MOVW	#24578, \$RMS_PTR	
			0094	C7		01	8E	0009C	MNEGB	#1, \$RMS_PTR+2	
			0096	C7		C7	9E	000A1	MOVAB	ACCOUNT_RSA_A, \$RMS_PTR+4	
			0098	C7	00F4	8F	28	000A8	MOVCS	#80, ACCOUNT_FAB_A, ACCOUNT_FAB_B	1415
01F4	C7			67	0050	8F	28	000B0	MOVCS	#68, ACCOUNT_RAB_A, ACCOUNT_RAB_B	1416
0244	C7		50	A7	0044	8F	28	000B9	MOVCS	#96, ACCOUNT_NAM_A, ACCOUNT_NAM_B	1417
0288	C7		0094	C7	0060	8F	28	000C3	MOVAB	ACCOUNT_NAM_B, ACCOUNT_FAB_B+40	1418
			021C	C7	0288	C7	9E	000CA	MOVAB	ACCOUNT_FAB_B, ACCOUNT_RAB_B+60	1419
			0280	C7	01F4	C7	9E	000D1	MOVAB	ACCOUNT_RSA_B, ACCOUNT_NAM_B+4	1420
			028C	C7	02E8	C7	9E	000D8	MOVL	ACCOUNTING_FABS, OLD_FAB	1426
				53	F0	A7	D0	000DC	MOVL	ACCOUNTING_FABS+4, NEW_FAB	1427
				52	F4	A7	D0	000E0	MOVL	ACCOUNTING_RABS, OLD_RAB	1428
				56	F8	A7	D0	000E4	MOVL	ACCOUNTING_RABS+4, NEW_RAB	1429
				55	FC	A7	D0	000E8	BISB2	#2, 7(NEW_FAB)	1436
			07	A2		02	88	000EC	MOVZWL	2(OLD_FAB), OLD_IF1	1437
				50	02	A3	3C	000F0	CLRL	R4	1438
						54	D4	000F2	TSTL	OLD_IF1	
						50	D5	000F4	BEQL	3\$	
						04	13	000F6	INCL	R4	
						54	D6	000F8	BRB	4\$	
				04	04	04	11	000FA	BLBC	NEW, 5\$	
				07	A2	02	8A	000FE	BICB2	#2, 7(NEW_FAB)	
					10	A2	D4	00102	CLRL	16(NEW_FAB)	1446
						52	DD	00105	PUSHL	NEW_FAB	1447
			00000000G	00		01	FB	00107	CALLS	#1, -SYSS\$CREATE	
				14		50	E8	0010E	BLBS	R0, 6\$	
EC			04	A2		19	E4	00111	BBSC	#25, 4(NEW_FAB), 5\$	1450
						52	DD	00116	PUSHL	NEW_FAB	1453
						52	DD	00118	PUSHL	NEW_FAB	
					000410A2	52	DD	0011A	PUSHL	#266402	
				68		8F	DD	00120	CALLS	#3, SIGNAL_FILE_ERROR	
						03	FB	00123	BRB	7\$	1452
						23	11	00125	PUSHL	NEW_RAB	1459
			00000000G	00		55	DD	00127	CALLS	#1, -SYSS\$CONNECT	
				17		01	FB	0012E	BLBS	R0, 7\$	
						50	E8	00131	PUSHR	#^M<R2,R5>	1462
						24	BB	00131			

		000410A2	8F	DD	00133	PUSHL	#266402	
	68		03	FB	00139	CALLS	#3, SIGNAL_FILE_ERROR	
			52	DD	0013C	PUSHL	NEW_FAB	1463
00000000G	00		01	FB	0013E	CALLS	#1, -SYSSCLOSE	
		02	A2	B4	00145	CLRW	2(NEW_FAB)	1464
0A	07	A2	01	E1	00148	BBC	#1, 7(NEW_FAB), 8\$	1475
00010619	8F	08	A2	D1	0014D	CMPL	8(NEW_FABT, #67097	
			15	12	00155	BNEQ	10\$	
	12		54	E9	00157	BLBC	R4, 10\$	1476
	09	04	AC	E8	0015A	BLBS	NEW, 9\$	1479
			52	DD	0015E	PUSHL	NEW_FAB	
			0D	DD	00160	PUSHL	#13-	
0000V	CF		02	FB	00162	CALLS	#2, WRITE_FILE_LINK_RECORD	
0000V	CF		00	FB	00167	CALLS	#0, CLOSE-ACCOUNTING_FILE	1480
F0	A7		52	7D	0016C	MOVQ	NEW_FAB, ACCOUNTING_FABS	1486
F8	A7		55	7D	00170	MOVQ	NEW_RAB, ACCOUNTING_RABS	1488
	09		54	E9	00174	BLBC	R4, 11\$	1494
			53	DD	00177	PUSHL	OLD_FAB	
0000V	CF		0E	DD	00179	PUSHL	#14-	
			02	FB	0017B	CALLS	#2, WRITE_FILE_LINK_RECORD	
			04	00180	11\$:	RET		1495

; Routine Size: 385 bytes, Routine Base: CODE + 0138

```
460 1496 1 GLOBAL ROUTINE CLOSE_ACCOUNTING_FILE: NOVALUE=
461 1497 1
462 1498 1 ++
463 1499 1
464 1500 1 FUNCTIONAL DESCRIPTION:
465 1501 1 This routine closes an accounting file, if one is open.
466 1502 1
467 1503 1 INPUT PARAMETERS:
468 1504 1 NONE
469 1505 1
470 1506 1 IMPLICIT INPUTS:
471 1507 1 NONE
472 1508 1
473 1509 1 OUTPUT PARAMETERS:
474 1510 1 NONE
475 1511 1
476 1512 1 IMPLICIT OUTPUTS:
477 1513 1 NONE
478 1514 1
479 1515 1 ROUTINE VALUE:
480 1516 1 NONE
481 1517 1
482 1518 1 SIDE EFFECTS:
483 1519 1 Accounting file closed.
484 1520 1
485 1521 1 --
486 1522 1
487 1523 2 BEGIN
488 1524 2 LOCAL
489 1525 2 FAB: REF BBLOCK, ! Pointer to FAB
490 1526 2 RAB: REF BBLOCK; ! Pointer to RAB
491 1527 2
492 1528 2
493 1529 2 FAB = .ACCOUNTING_FABS[0];
494 1530 2 RAB = .ACCOUNTING_RABS[0];
495 1531 2
496 1532 2
497 1533 2 IF .FAB[FAB$W_IFI] NEQ 0
498 1534 2 THEN
499 1535 3 BEGIN
500 1536 3 IF TESTBITSC(RAB[RAB$V_ASY])
501 1537 3 THEN
502 1538 4 IF NOT $WAIT(RAB=.RAB)
503 1539 3 THEN
504 1540 3 SIGNAL_FILE_ERROR(JBC$_WRITEERR + STS$K_ERROR, .FAB, .RAB);
505 1541 3
506 1542 3
507 1543 3 IF NOT $CLOSE(FAB=.FAB)
508 1544 3 THEN
509 1545 3 SIGNAL_FILE_ERROR(JBC$_CLOSEOUT + STS$K_ERROR, .FAB, .FAB);
510 1546 3
511 1547 3
512 1548 3 FAB[FAB$W_IFI] = 0;
513 1549 2 END;
514 1550 1 END;
```

			001C 00000	.ENTRY	CLOSE ACCOUNTING FILE, Save R2,R3,R4	: 1496
	54	00000000G	EF 9E 00002	MOVAB	SIGNAL_FILE_ERROR, R4	: 1529
	53	00000000'	EF D0 00009	MOVL	ACCOUNTING_FABS, FAB	: 1530
	52	00000000'	EF D0 00010	MOVL	ACCOUNTING_RABS, RAB	: 1533
		02	A3 B5 00017	TSTW	2(FAB)	: 1536
19	04	A2	3A 13 0001A	BEQL	3\$: 1538
			00 E5 0001C	BBCC	#0, 4(RAB), 1\$: 1540
	00000000G	00	52 DD 00021	PUSHL	RAB	: 1543
		0D	01 FB 00023	CALLS	#1, SYSSWAIT	: 1545
			50 E8 0002A	BLBS	R0, 1\$: 1548
			52 DD 0002D	PUSHL	RAB	: 1550
			53 DD 0002F	PUSHL	FAB	: 1553
		000410D2	8F DD 00031	PUSHL	#266450	: 1556
	64		03 FB 00037	CALLS	#3, SIGNAL_FILE_ERROR	: 1559
			53 DD 0003A 1\$:	PUSHL	FAB	: 1562
	00000000G	00	01 FB 0003C	CALLS	#1, SYSCLOSE	: 1565
		0D	50 E8 00043	BLBS	R0, 2\$: 1568
			53 DD 00046	PUSHL	FAB	: 1571
			53 DD 00048	PUSHL	FAB	: 1574
		0004105A	8F DD 0004A	PUSHL	#266330	: 1577
	64		03 FB 00050	CALLS	#3, SIGNAL_FILE_ERROR	: 1580
		02	A3 B4 00053 2\$:	CLRW	2(FAB)	: 1583
			04 00056 3\$:	RET		: 1586

; Routine Size: 87 bytes, Routine Base: CODE + 02B9

```
516 M 1551 1 MACRO ACM_RECORD(TYPE,SUBTYPE,TIME,ACR)=
517 M 1552 1
518 M 1553 1 ++
519 M 1554 1
520 M 1555 1 FUNCTIONAL DESCRIPTION:
521 M 1556 1 This macro builds the record header.
522 M 1557 1
523 M 1558 1 INPUT PARAMETERS:
524 M 1559 1 TYPE - Record type.
525 M 1560 1 SUBTYPE - Record subtype.
526 M 1561 1 TIME - Pointer to quadword event time.
527 M 1562 1 ACR - Pointer to accounting record buffer.
528 M 1563 1
529 M 1564 1 IMPLICIT INPUTS:
530 M 1565 1 NONE
531 M 1566 1
532 M 1567 1 OUTPUT PARAMETERS:
533 M 1568 1 NONE
534 M 1569 1
535 M 1570 1 IMPLICIT OUTPUTS:
536 M 1571 1 Record header built in record buffer.
537 M 1572 1
538 M 1573 1 SIDE EFFECTS:
539 M 1574 1 NONE
540 M 1575 1
541 M 1576 1 --
542 M 1577 1
543 M 1578 1 BEGIN
544 M 1579 1 BBLOCK[ACR, ACR$W_LENGTH] = ACR$K_HDRLEN;
545 M 1580 1 %IF %CTCE(TYPE) AND %CTCE(SUBTYPE)
546 M 1581 1 %THEN
547 M 1582 1 BBLOCK[ACR, ACR$W_TYPE] =
548 M 1583 1 (TYPE) ^ $BITPOSITION(ACR$V_TYPE) OR
549 M 1584 1 (SUBTYPE) ^ $BITPOSITION(ACR$V_SUBTYPE) OR
550 M 1585 1 ACR$K_CURVER ^ $BITPOSITION(ACR$V_VERSION)
551 M 1586 1 %ELSE
552 M 1587 1 %IF %CTCE(SUBTYPE)
553 M 1588 1 %THEN
554 M 1589 1 BBLOCK[ACR, ACR$W_TYPE] =
555 M 1590 1 (SUBTYPE) ^ $BITPOSITION(ACR$V_SUBTYPE) OR
556 M 1591 1 ACR$K_CURVER ^ $BITPOSITION(ACR$V_VERSION);
557 M 1592 1 BBLOCK[ACR, ACR$V_TYPE] = (TYPE)
558 M 1593 1 %ELSE
559 M 1594 1 BBLOCK[ACR, ACR$W_TYPE] =
560 M 1595 1 ACR$K_CURVER ^ $BITPOSITION(ACR$V_VERSION);
561 M 1596 1 BBLOCK[ACR, ACR$V_TYPE] = (TYPE);
562 M 1597 1 BBLOCK[ACR, ACR$V_SUBTYPE] = (SUBTYPE)
563 M 1598 1 %FI
564 M 1599 1 %FI;
565 M 1600 1 (BBLOCK[ACR, ACR$Q_SYSTIME]+0) = .VECTOR[TIME, 0];
566 M 1601 1 (BBLOCK[ACR, ACR$Q_SYSTIME]+4) = .VECTOR[TIME, 1];
567 M 1602 1 END %;
```

```
569 M 1603 1 MACRO ACM_PACKET(TYPE,SUBTYPE,ACR,APK)=
570 M 1604 1
571 M 1605 1 ++
572 M 1606 1
573 M 1607 1 FUNCTIONAL DESCRIPTION:
574 M 1608 1 This macro builds the packet header.
575 M 1609 1
576 M 1610 1 INPUT PARAMETERS:
577 M 1611 1 TYPE - Packet type.
578 M 1612 1 SUBTYPE - Packet subtype.
579 M 1613 1 ACR - Pointer to accounting record buffer.
580 M 1614 1
581 M 1615 1 IMPLICIT INPUTS:
582 M 1616 1 NONE
583 M 1617 1
584 M 1618 1 OUTPUT PARAMETERS:
585 M 1619 1 APK - Pointer to packet.
586 M 1620 1
587 M 1621 1 IMPLICIT OUTPUTS:
588 M 1622 1 Packet header built in record buffer.
589 M 1623 1
590 M 1624 1 SIDE EFFECTS:
591 M 1625 1 NONE
592 M 1626 1
593 M 1627 1 --
594 M 1628 1
595 M 1629 1 BEGIN
596 M 1630 1 APK = (ACR) + .BBLOCK[ACR, ACR$W_LENGTH];
597 M 1631 1 APK[ACR$W_LENGTH] = 0;
598 M 1632 1 %IF %CTCE(TYPE) AND %CTCE(SUBTYPE)
599 M 1633 1 %THEN
600 M 1634 1 APK[ACR$W_TYPE] =
601 M 1635 1 ACR$M_PACKET OR
602 M 1636 1 (TYPE) * $BITPOSITION(ACR$V_TYPE) OR
603 M 1637 1 (SUBTYPE) * $BITPOSITION(ACR$V_SUBTYPE) OR
604 M 1638 1 ACR$K_CURVER * $BITPOSITION(ACR$V_VERSION)
605 M 1639 1 %ELSE
606 M 1640 1 APK[ACR$W_TYPE] =
607 M 1641 1 ACR$M_PACKET OR
608 M 1642 1 ACR$K_CURVER * $BITPOSITION(ACR$V_VERSION);
609 M 1643 1 APK[ACR$V_TYPE] = (TYPE);
610 M 1644 1 APK[ACR$V_SUBTYPE] = (SUBTYPE)
611 M 1645 1 %FI;
612 M 1646 1 END %;
```

```
614 1647 1 ROUTINE IDENT_PACKET: L_IDENT_PACKET NOVALUE=
615 1648 1
616 1649 1 ++
617 1650 1
618 1651 1 FUNCTIONAL DESCRIPTION:
619 1652 1 This routine builds the identification packet.
620 1653 1
621 1654 1 INPUT PARAMETERS:
622 1655 1 NONE
623 1656 1
624 1657 1 IMPLICIT INPUTS:
625 1658 1 ACM - Pointer to mailbox message.
626 1659 1 ACR - Pointer to accounting record buffer.
627 1660 1 SJH - Pointer to SJH or 0.
628 1661 1 SMQ - Pointer to SMQ or 0.
629 1662 1
630 1663 1 OUTPUT PARAMETERS:
631 1664 1 NONE
632 1665 1
633 1666 1 IMPLICIT OUTPUTS:
634 1667 1 Identification packet built in record buffer.
635 1668 1
636 1669 1 ROUTINE VALUE:
637 1670 1 NONE
638 1671 1
639 1672 1 SIDE EFFECTS:
640 1673 1 NONE
641 1674 1
642 1675 1 --
643 1676 1
644 1677 2 BEGIN
645 1678 2 EXTERNAL REGISTER
646 1679 2 ACM = ACCT_ACM_REG: REF BBLOCK, ! Pointer to mailbox message
647 1680 2 ACR = ACCT_ACR_REG: REF BBLOCK, ! Pointer to record buffer
648 1681 2 SJH = ACCT_SJH_REG: REF BBLOCK, ! Pointer to SJH
649 1682 2 SMQ = ACCT_SMQ_REG: REF BBLOCK, ! Pointer to SMQ
650 1683 2 LOCAL
651 1684 2 APK: REF BBLOCK; ! Pointer to packet
652 1685 2 REGISTER
653 1686 2 P = 3; ! Pointer to free byte
654 1687 2
655 1688 2
656 1689 2 ACM_PACKET(ACR&K_ID, 0, .ACR, APK);
657 1690 2 APK[ACR&L_PID] = .ACM[ACM&L_PID];
658 1691 2 APK[ACR&L_OWNER] = .ACM[ACM&L_OWNER];
659 1692 2 APK[ACR&L_UIC] = .ACM[ACM&L_UIC];
660 1693 2 (APK[ACR&Q_PRIV]+0) = .(ACM[ACM&Q_PRIVMSK]+0);
661 1694 2 (APK[ACR&Q_PRIV]+4) = .(ACM[ACM&Q_PRIVMSK]+4);
662 1695 2 APK[ACR&B_PRI] = .ACM[ACM&B_PROCPRI];
663 1696 2 (APK[ACR&B_PRI]+1)<0,8> = 0;
664 1697 2
665 1698 2
666 1699 2 APK[ACR&W_ACCOUNT] = 0;
667 1700 2 APK[ACR&W_NODENAME] = 0;
668 1701 2 APK[ACR&W_TERMINAL] = 0;
669 1702 2 APK[ACR&W_JOBNAME] = 0;
670 1703 2 APK[ACR&L_JOBID] = 0;
```

```
671 1704 2 APK[ACRSW_QUEUE] = 0;
672 1705 2 APK[ACRSW_NODEADDR] = 0;
673 1706 2 APK[ACRSW_REMOTEID] = 0;
674 1707 2 P = APK[ACRSK_IDVAR,0,0,0];
675 1708
676 1709
677 1710 BEGIN ! block to use output registers
678 1711 REGISTER
679 1712 RO = 0;
680 1713
681 1714 APK[ACRSW_USERNAME] = .P - .APK;
682 1715 LOCC(%REF(%C' '), %REF(ACM$$_USERNAME), ACM[ACM$$_USERNAME]; RO);
683 1716 RO = ACM$$_USERNAME - .RO;
684 1717 (.P)<0,8> = .RO;
685 1718 P = .P + 1;
686 1719 MOVCL(RO, ACM[ACM$$_USERNAME], .P; ... P);
687 1720 END; ! block to use output registers
688 1721
689 1722
690 1723 BEGIN ! block to use output registers
691 1724 REGISTER
692 1725 RO = 0;
693 1726 R1 = 1 : REF VECTOR[.BYTE];
694 1727
695 1728
696 1729 ! Strip leading binary nulls and trailing blanks from the account name.
697 1730 ! Don't move over the account name at all if it is totally binary nulls.
698 1731
699 1732 IF SKPC(%REF(0), %REF(ACM$$_ACCOUNT), ACM[ACM$$_ACCOUNT]; RO, R1)
700 1733 THEN
701 1734 BEGIN
702 1735 DO
703 1736 BEGIN
704 1737 IF .R1[.RO - 1] NEQ %C' '
705 1738 THEN
706 1739 EXITLOOP;
707 1740 RO = .RO - 1;
708 1741 END
709 1742 WHILE .RO GTR 0;
710 1743 APK[ACRSW_ACCOUNT] = .P - .APK;
711 1744 (.P)<0,8> = .RO;
712 1745 P = .P + 1;
713 1746 MOVCL(RO, .R1, .P; ... P);
714 1747 END;
715 1748 END; ! block to use output registers
716 1749
717 1750
718 P 1751 IF ONEOF (.ACR[ACRSV_TYPE], BMSK_ (
719 1752 ACRSK_PRCDEL, ACRSK_LOGFAIL, ACRSK_IMGDEL, ACRSK_PRCPUR, ACRSK_IMGPUR))
720 1753 THEN
721 1754 BEGIN
722 1755 IF .ACM[ACMSW_NODEADDR] NEQ 0
723 1756 THEN
724 1757 BEGIN
725 1758 LOCAL
726 1759 Q: REF VECTOR[.BYTE], ! Pointer to ASCII data
727 1760 L: ! Length of data
```

```
728      1761 4
729      1762 4
730      1763 4      Q = .ACM + .ACM[ACMSW_NODEADDR];
731      1764 4      L = .Q[0];
732      1765 4      IF .L NEQ 0
733      1766 5      THEN
734      1767 5          BEGIN
735      1768 5              APK[ACRSW_NODEADDR] = .P - .APK;
736      1769 4              MOV3(%REF(.L + 1), .Q, .P; ..., P);
737      1770 4              END;
738      1771 4      END;
739      1772 4
740      1773 4      IF .ACM[ACMSW_NODENAME] NEQ 0
741      1774 4      THEN
742      1775 5          BEGIN
743      1776 5              LOCAL
744      1777 4                  Q:          REF VECTOR[BYTE],      ! Pointer to ASCII data
745      1778 4                  L:          ! Length of data
746      1779 4
747      1780 4              Q = .ACM + .ACM[ACMSW_NODENAME];
748      1781 4              L = .Q[0];
749      1782 4              IF .L NEQ 0
750      1783 4              THEN
751      1784 5                  BEGIN
752      1785 5                      APK[ACRSW_NODENAME] = .P - .APK;
753      1786 5                      MOV3(%REF(.L + 1), .Q, .P; ..., P);
754      1787 4                      END;
755      1788 4                  END;
756      1789 4
757      1790 4      IF .ACM[ACMSW_REMOTEID] NEQ 0
758      1791 4      THEN
759      1792 5          BEGIN
760      1793 5              LOCAL
761      1794 4                  Q:          REF VECTOR[BYTE],      ! Pointer to ASCII data
762      1795 4                  L:          ! Length of data
763      1796 4
764      1797 4              Q = .ACM + .ACM[ACMSW_REMOTEID];
765      1798 4              L = .Q[0];
766      1799 4              IF .L NEQ 0
767      1800 4              THEN
768      1801 5                  BEGIN
769      1802 5                      APK[ACRSW_REMOTEID] = .P - .APK;
770      1803 5                      MOV3(%REF(.L + 1), .Q, .P; ..., P);
771      1804 4                      END;
772      1805 4                  END;
773      1806 4              END;
774      1807 4      END;
775      1808 4
776      1809 4      IF CH$RCHAR(ACM[ACM$T_TERMINAL]) NEQ 0
777      1810 4      THEN
778      1811 5          BEGIN
779      1812 5              APK[ACRSW_TERMINAL] = .P - .APK;
780      1813 5              MOV3(%REF(CH$RCHAR(ACM[ACM$T_TERMINAL]) + 1), ACM[ACM$T_TERMINAL], .P; ..., P);
781      1814 4              END;
782      1815 4
783      1816 4
784      1817 4
```

```
785 1818 2 IF .SJH NEQ 0
786 1819 THEN
787 1820 BEGIN
788 1821   APK[ACR$J-JOBID] = .SJH[SYMSL_ENTRY_NUMBER];
789 1822   APK[ACR$W-JOBNAME] = .P - .APK;
790 1823   MOV3(%REF(CH$RCHAR(SJH[SJH$T_NAME]) + 1), SJH[SJH$T_NAME], .P; ... P);
791 1824   APK[ACR$W-QUEUE] = .P - .APK;
792 1825   MOV3(%REF(CH$RCHAR(SMQ[SMQ$T_NAME]) + 1), SMQ[SMQ$T_NAME], .P; ... P);
793 1826 END;
794 1827
795 1828
796 1829   APK[ACR$W-LENGTH] = .P - .APK;
797 1830   ACR[ACR$W-LENGTH] = .P - .ACR;
798 1831 1 END;
```

```
013C 00000 IDENT_PACKET:
                                .WORD      Save R2,R3,R4,R5,R8
                                MOVZWL     2(ACR), APK
                                ADDL2      ACR, APK
                                MOVZWL     #8195, (APK)
                                MOVL       40(ACM), 4(APK)
                                MOVL       48(ACM), 8(APK)
                                MOVL       12(ACM), 12(APK)
                                MOVQ       4(ACM), 16(APK)
                                MOVZBW     36(ACM), 24(APK)
                                CLRL       28(APK)
                                CLRQ       32(APK)
                                CLRL       40(APK)
                                CLRW       44(APK)
                                MOVAB      46(R8), P
                                SUBW3      APK, P, 26(APK)
                                LOCC       #32, #12, 16(ACM)
                                SUBL3      R0, #12, R0
                                MOVB       R0, (P)+
                                MOV3       R0, 16(ACM), (P)
                                SKPC       #0, #8, 28(ACM)
                                BEQL       3$
                                CMPB       -1(R0)[R1], #32
                                BNEQ       2$
                                SOBGTR     R0, 1$
                                SUBW3      APK, P, 28(APK)
                                MOVB       R0, (P)+
                                MOV3       R0, (R1), (P)
                                EXTZV      #1, #7, (ACR), R0
                                ASHL       R0, #2030043136, R0
                                BGEQ       6$
                                TSTW       116(ACM)
                                BEQL       4$
                                MOVZWL     116(ACM), Q
                                ADDL2      ACM, Q
                                MOVZBL     (Q), L
                                BEQL       4$
                                SUBW3      APK, P, 42(APK)

50 1A  A8  53  02  A7  3C  00002  1$:
10  A6  0C  02  57  C0  00006  2$:
   50  0C  03  8F  3C  00009  3$:
   63  10  28  A6  D0  0000E  1$:
   67  10  30  A6  D0  00013  2$:
   79  10  0C  A6  D0  00018  3$:
   83  10  04  A6  7D  0001D  1$:
   87  10  24  A6  9B  00022  2$:
   91  10  1C  A8  D4  00027  3$:
   95  10  20  A8  7C  0002A  1$:
   99  10  28  A8  D4  0002D  2$:
  103  10  2C  A8  B4  00030  3$:
  107  10  2E  A8  9E  00033  1$:
  111  10  53  58  A3  00037  2$:
  115  10  0C  20  3A  0003C  3$:
  119  10  0C  50  C3  00041  1$:
  123  10  83  50  90  00045  2$:
  127  10  63  50  28  00048  3$:
  131  10  A6  00  3B  0004D  1$:
  135  10  08  16  13  00052  2$:
  139  10  20  FF  A0  41  91  00054  3$:
  143  10  F6  03  12  00059  1$:
  147  10  53  50  F5  0005B  2$:
  151  10  83  58  A3  0005E  3$:
  155  10  61  50  90  00063  1$:
  159  10  07  50  28  00066  2$:
  163  10  8F  01  EF  0006A  3$:
  167  10  50  50  78  0006F  1$:
  171  10  74  54  1B  00077  2$:
  175  10  17  A6  B5  00079  3$:
  179  10  51  17  13  0007C  1$:
  183  10  51  74  A6  3C  0007E  2$:
  187  10  50  56  C0  00082  3$:
  191  10  08  61  9A  00085  1$:
  195  10  58  0B  13  00088  2$:
  199  10  2A  A8  53  58  A3  0008A  3$:
  203  10  50  50  78  0006F  1$:
  207  10  74  54  1B  00077  2$:
  211  10  17  A6  B5  00079  3$:
  215  10  51  17  13  0007C  1$:
  219  10  51  74  A6  3C  0007E  2$:
  223  10  50  56  C0  00082  3$:
  227  10  08  61  9A  00085  1$:
  231  10  58  0B  13  00088  2$:
  235  10  2A  A8  53  58  A3  0008A  3$:
  239  10  50  50  78  0006F  1$:
  243  10  74  54  1B  00077  2$:
  247  10  17  A6  B5  00079  3$:
  251  10  51  17  13  0007C  1$:
  255  10  51  74  A6  3C  0007E  2$:
  259  10  50  56  C0  00082  3$:
  263  10  08  61  9A  00085  1$:
  267  10  58  0B  13  00088  2$:
  271  10  2A  A8  53  58  A3  0008A  3$:
  275  10  50  50  78  0006F  1$:
  279  10  74  54  1B  00077  2$:
  283  10  17  A6  B5  00079  3$:
  287  10  51  17  13  0007C  1$:
  291  10  51  74  A6  3C  0007E  2$:
  295  10  50  56  C0  00082  3$:
  299  10  08  61  9A  00085  1$:
  303  10  58  0B  13  00088  2$:
  307  10  2A  A8  53  58  A3  0008A  3$:
  311  10  50  50  78  0006F  1$:
  315  10  74  54  1B  00077  2$:
  319  10  17  A6  B5  00079  3$:
  323  10  51  17  13  0007C  1$:
  327  10  51  74  A6  3C  0007E  2$:
  331  10  50  56  C0  00082  3$:
  335  10  08  61  9A  00085  1$:
  339  10  58  0B  13  00088  2$:
  343  10  2A  A8  53  58  A3  0008A  3$:
  347  10  50  50  78  0006F  1$:
  351  10  74  54  1B  00077  2$:
  355  10  17  A6  B5  00079  3$:
  359  10  51  17  13  0007C  1$:
  363  10  51  74  A6  3C  0007E  2$:
  367  10  50  56  C0  00082  3$:
  371  10  08  61  9A  00085  1$:
  375  10  58  0B  13  00088  2$:
  379  10  2A  A8  53  58  A3  0008A  3$:
  383  10  50  50  78  0006F  1$:
  387  10  74  54  1B  00077  2$:
  391  10  17  A6  B5  00079  3$:
  395  10  51  17  13  0007C  1$:
  399  10  51  74  A6  3C  0007E  2$:
  403  10  50  56  C0  00082  3$:
  407  10  08  61  9A  00085  1$:
  411  10  58  0B  13  00088  2$:
  415  10  2A  A8  53  58  A3  0008A  3$:
  419  10  50  50  78  0006F  1$:
  423  10  74  54  1B  00077  2$:
  427  10  17  A6  B5  00079  3$:
  431  10  51  17  13  0007C  1$:
  435  10  51  74  A6  3C  0007E  2$:
  439  10  50  56  C0  00082  3$:
  443  10  08  61  9A  00085  1$:
  447  10  58  0B  13  00088  2$:
  451  10  2A  A8  53  58  A3  0008A  3$:
  455  10  50  50  78  0006F  1$:
  459  10  74  54  1B  00077  2$:
  463  10  17  A6  B5  00079  3$:
  467  10  51  17  13  0007C  1$:
  471  10  51  74  A6  3C  0007E  2$:
  475  10  50  56  C0  00082  3$:
  479  10  08  61  9A  00085  1$:
  483  10  58  0B  13  00088  2$:
  487  10  2A  A8  53  58  A3  0008A  3$:
  491  10  50  50  78  0006F  1$:
  495  10  74  54  1B  00077  2$:
  499  10  17  A6  B5  00079  3$:
  503  10  51  17  13  0007C  1$:
  507  10  51  74  A6  3C  0007E  2$:
  511  10  50  56  C0  00082  3$:
  515  10  08  61  9A  00085  1$:
  519  10  58  0B  13  00088  2$:
  523  10  2A  A8  53  58  A3  0008A  3$:
  527  10  50  50  78  0006F  1$:
  531  10  74  54  1B  00077  2$:
  535  10  17  A6  B5  00079  3$:
  539  10  51  17  13  0007C  1$:
  543  10  51  74  A6  3C  0007E  2$:
  547  10  50  56  C0  00082  3$:
  551  10  08  61  9A  00085  1$:
  555  10  58  0B  13  00088  2$:
  559  10  2A  A8  53  58  A3  0008A  3$:
  563  10  50  50  78  0006F  1$:
  567  10  74  54  1B  00077  2$:
  571  10  17  A6  B5  00079  3$:
  575  10  51  17  13  0007C  1$:
  579  10  51  74  A6  3C  0007E  2$:
  583  10  50  56  C0  00082  3$:
  587  10  08  61  9A  00085  1$:
  591  10  58  0B  13  00088  2$:
  595  10  2A  A8  53  58  A3  0008A  3$:
  599  10  50  50  78  0006F  1$:
  603  10  74  54  1B  00077  2$:
  607  10  17  A6  B5  00079  3$:
  611  10  51  17  13  0007C  1$:
  615  10  51  74  A6  3C  0007E  2$:
  619  10  50  56  C0  00082  3$:
  623  10  08  61  9A  00085  1$:
  627  10  58  0B  13  00088  2$:
  631  10  2A  A8  53  58  A3  0008A  3$:
  635  10  50  50  78  0006F  1$:
  639  10  74  54  1B  00077  2$:
  643  10  17  A6  B5  00079  3$:
  647  10  51  17  13  0007C  1$:
  651  10  51  74  A6  3C  0007E  2$:
  655  10  50  56  C0  00082  3$:
  659  10  08  61  9A  00085  1$:
  663  10  58  0B  13  00088  2$:
  667  10  2A  A8  53  58  A3  0008A  3$:
  671  10  50  50  78  0006F  1$:
  675  10  74  54  1B  00077  2$:
  679  10  17  A6  B5  00079  3$:
  683  10  51  17  13  0007C  1$:
  687  10  51  74  A6  3C  0007E  2$:
  691  10  50  56  C0  00082  3$:
  695  10  08  61  9A  00085  1$:
  699  10  58  0B  13  00088  2$:
  703  10  2A  A8  53  58  A3  0008A  3$:
  707  10  50  50  78  0006F  1$:
  711  10  74  54  1B  00077  2$:
  715  10  17  A6  B5  00079  3$:
  719  10  51  17  13  0007C  1$:
  723  10  51  74  A6  3C  0007E  2$:
  727  10  50  56  C0  00082  3$:
  731  10  08  61  9A  00085  1$:
  735  10  58  0B  13  00088  2$:
  739  10  2A  A8  53  58  A3  0008A  3$:
  743  10  50  50  78  0006F  1$:
  747  10  74  54  1B  00077  2$:
  751  10  17  A6  B5  00079  3$:
  755  10  51  17  13  0007C  1$:
  759  10  51  74  A6  3C  0007E  2$:
  763  10  50  56  C0  00082  3$:
  767  10  08  61  9A  00085  1$:
  771  10  58  0B  13  00088  2$:
  775  10  2A  A8  53  58  A3  0008A  3$:
  779  10  50  50  78  0006F  1$:
  783  10  74  54  1B  00077  2$:
  787  10  17  A6  B5  00079  3$:
  791  10  51  17  13  0007C  1$:
  795  10  51  74  A6  3C  0007E  2$:
  799  10  50  56  C0  00082  3$:
  803  10  08  61  9A  00085  1$:
  807  10  58  0B  13  00088  2$:
  811  10  2A  A8  53  58  A3  0008A  3$:
  815  10  50  50  78  0006F  1$:
  819  10  74  54  1B  00077  2$:
  823  10  17  A6  B5  00079  3$:
  827  10  51  17  13  0007C  1$:
  831  10  51  74  A6  3C  0007E  2$:
  835  10  50  56  C0  00082  3$:
  839  10  08  61  9A  00085  1$:
  843  10  58  0B  13  00088  2$:
  847  10  2A  A8  53  58  A3  0008A  3$:
  851  10  50  50  78  0006F  1$:
  855  10  74  54  1B  00077  2$:
  859  10  17  A6  B5  00079  3$:
  863  10  51  17  13  0007C  1$:
  867  10  51  74  A6  3C  0007E  2$:
  871  10  50  56  C0  00082  3$:
  875  10  08  61  9A  00085  1$:
  879  10  58  0B  13  00088  2$:
  883  10  2A  A8  53  58  A3  0008A  3$:
  887  10  50  50  78  0006F  1$:
  891  10  74  54  1B  00077  2$:
  895  10  17  A6  B5  00079  3$:
  899  10  51  17  13  0007C  1$:
  903  10  51  74  A6  3C  0007E  2$:
  907  10  50  56  C0  00082  3$:
  911  10  08  61  9A  00085  1$:
  915  10  58  0B  13  00088  2$:
  919  10  2A  A8  53  58  A3  0008A  3$:
  923  10  50  50  78  0006F  1$:
  927  10  74  54  1B  00077  2$:
  931  10  17  A6  B5  00079  3$:
  935  10  51  17  13  0007C  1$:
  939  10  51  74  A6  3C  0007E  2$:
  943  10  50  56  C0  00082  3$:
  947  10  08  61  9A  00085  1$:
  951  10  58  0B  13  00088  2$:
  955  10  2A  A8  53  58  A3  0008A  3$:
  959  10  50  50  78  0006F  1$:
  963  10  74  54  1B  00077  2$:
  967  10  17  A6  B5  00079  3$:
  971  10  51  17  13  0007C  1$:
  975  10  51  74  A6  3C  0007E  2$:
  979  10  50  56  C0  00082  3$:
  983  10  08  61  9A  00085  1$:
  987  10  58  0B  13  00088  2$:
  991  10  2A  A8  53  58  A3  0008A  3$:
  995  10  50  50  78  0006F  1$:
  999  10  74  54  1B  00077  2$:
  1003 10  17  A6  B5  00079  3$:
  1007 10  51  17  13  0007C  1$:
  1011 10  51  74  A6  3C  0007E  2$:
  1015 10  50  56  C0  00082  3$:
  1019 10  08  61  9A  00085  1$:
  1023 10  58  0B  13  00088  2$:
  1027 10  2A  A8  53  58  A3  0008A  3$:
  1031 10  50  50  78  0006F  1$:
  1035 10  74  54  1B  00077  2$:
  1039 10  17  A6  B5  00079  3$:
  1043 10  51  17  13  0007C  1$:
  1047 10  51  74  A6  3C  0007E  2$:
  1051 10  50  56  C0  00082  3$:
  1055 10  08  61  9A  00085  1$:
  1059 10  58  0B  13  00088  2$:
  1063 10  2A  A8  53  58  A3  0008A  3$:
  1067 10  50  50  78  0006F  1$:
  1071 10  74  54  1B  00077  2$:
  1075 10  17  A6  B5  00079  3$:
  1079 10  51  17  13  0007C  1$:
  1083 10  51  74  A6  3C  0007E  2$:
  1087 10  50  56  C0  00082  3$:
  1091 10  08  61  9A  00085  1$:
  1095 10  58  0B  13  00088  2$:
  1099 10  2A  A8  53  58  A3  0008A  3$:
  1103 10  50  50  78  0006F  1$:
  1107 10  74  54  1B  00077  2$:
  1111 10  17  A6  B5  00079  3$:
  1115 10  51  17  13  0007C  1$:
  1119 10  51  74  A6  3C  0007E  2$:
  1123 10  50  56  C0  00082  3$:
  1127 10  08  61  9A  00085  1$:
  1131 10  58  0B  13  00088  2$:
  1135 10  2A  A8  53  58  A3  0008A  3$:
  1139 10  50  50  78  0006F  1$:
  1143 10  74  54  1B  00077  2$:
  1147 10  17  A6  B5  00079  3$:
  1151 10  51  17  13  0007C  1$:
  1155 10  51  74  A6  3C  0007E  2$:
  1159 10  50  56  C0  00082  3$:
  1163 10  08  61  9A  00085  1$:
  1167 10  58  0B  13  00088  2$:
  1171 10  2A  A8  53  58  A3  0008A  3$:
  1175 10  50  50  78  0006F  1$:
  1179 10  74  54  1B  00077  2$:
  1183 10  17  A6  B5  00079  3$:
  1187 10  51  17  13  0007C  1$:
  1191 10  51  74  A6  3C  0007E  2$:
  1195 10  50  56  C0  00082  3$:
  1199 10  08  61  9A  00085  1$:
  1203 10  58  0B  13  00088  2$:
  1207 10  2A  A8  53  58  A3  0008A  3$:
  1211 10  50  50  78  0006F  1$:
  1215 10  74  54  1B  00077  2$:
  1219 10  17  A6  B5  00079  3$:
  1223 10  51  17  13  0007C  1$:
  1227 10  51  74  A6  3C  0007E  2$:
  1231 10  50  56  C0  00082  3$:
  1235 10  08  61  9A  00085  1$:
  1239 10  58  0B  13  00088  2$:
  1243 10  2A  A8  53  58  A3  0008A  3$:
  1247 10  50  50  78  0006F  1$:
  1251 10  74  54  1B  00077  2$:
  1255 10  17  A6  B5  00079  3$:
  1259 10  51  17  13  0007C  1$:
  1263 10  51  74  A6  3C  0007E  2$:
  1267 10  50  56  C0  00082  3$:
  1271 10  08  61  9A  00085  1$:
  1275 10  58  0B  13  00088  2$:
  1279 10  2A  A8  53  58  A3  0008A  3$:
  1283 10  50  50  78  0006F  1$:
  1287 10  74  54  1B  00077  2$:
  1291 10  17  A6  B5  00079  3$:
  1295 10  51  17  13  0007C  1$:
  1299 10  51  74  A6  3C  0007E  2$:
  1303 10  50  56  C0  00082  3$:
  1307 10  08  61  9A  00085  1$:
  1311 10  58  0B  13  00088  2$:
  1315 10  2A  A8  53  58  A3  0008A  3$:
  1319 10  50  50  78  0006F  1$:
  1323 10  74  54  1B  00077  2$:
  1327 10  17  A6  B5  00079  3$:
  1331 10  51  17  13  0007C  1$:
  1335 10  51  74  A6  3C  0007E  2$:
  1339 10  50  56  C0  00082  3$:
  1343 10  08  61  9A  00085  1$:
  1347 10  58  0B  13  00088  2$:
  1351 10  2A  A8  53  58  A3  0008A  3$:
  1355 10  50  50  78  0006F  1$:
  1359 10  74  54  1B  00077  2$:
  1363 10  17  A6  B5  00079  3$:
  1367 10  51  17  13  0007C  1$:
  1371 10  51  74  A6  3C  0007E  2$:
  1375 10  50  56  C0  00082  3$:
  1379 10  08  61  9A  00085  1$:
  1383 10  58  0B  13  00088  2$:
  1387 10  2A  A8  53  58  A3  0008A  3$:
  1391 10  50  50  78  0006F  1$:
  1395 10  74  54  1B  00077  2$:
  1399 10  17  A
```

				50	D6	0008F		INCL	R0		1768
	63		61	50	28	00091		MOV3	R0, (Q), (P)		
				76	A6	B5 00095	4\$:	TSTW	118(ACM)		1773
				17	13	00098		BEQL	5\$		
			51	76	A6	3C 0009A		MOVZWL	118(ACM), Q		1780
			51		56	C0 0009E		ADDL2	ACM, Q		
			50		61	9A 000A1		MOVZBL	(Q), L		1781
					0B	13 000A4		BEQL	5\$		1782
1E	A8		53		58	A3 000A6		SUBW3	APK, P, 30(APK)		1785
					50	D6 000AB		INCL	R0		1786
	63		61		50	28 000AD		MOV3	R0, (Q), (P)		
				78	A6	B5 000B1	5\$:	TSTW	120(ACM)		1791
				17	13	000B4		BEQL	6\$		
			51	78	A6	3C 000B6		MOVZWL	120(ACM), Q		1798
			51		56	C0 000BA		ADDL2	ACM, Q		
			50		61	9A 000BD		MOVZBL	(Q), L		1799
					0B	13 000C0		BEQL	6\$		1800
2C	A8		53		58	A3 000C2		SUBW3	APK, P, 44(APK)		1803
					50	D6 000C7		INCL	R0		1804
	63		61		50	28 000C9		MOV3	R0, (Q), (P)		
				34	A6	95 000CD	6\$:	TSTB	52(ACM)		1810
				10	13	000D0		BEQL	7\$		
20	A8		53		58	A3 000D2		SUBW3	APK, P, 32(APK)		1813
			50	34	A6	9A 000D7		MOVZBL	52(ACM), R0		1814
					50	D6 000DB		INCL	R0		
	63	34	A6		50	28 000DD		MOV3	R0, 52(ACM), (P)		
					59	D5 000E2	7\$:	TSTL	SJH		1818
				29	13	000E4		BEQL	8\$		
		24	A8	08	A9	D0 000E6		MOVL	8(SJH), 36(APK)		1821
22	A8		53		58	A3 000EB		SUBW3	APK, P, 34(APK)		1822
			50	0108	C9	9A 000F0		MOVZBL	264(SJH), R0		1823
					50	D6 000F5		INCL	R0		
	63	0108	C9		50	28 000F7		MOV3	R0, 264(SJH), (P)		
28	A8		53		58	A3 000FD		SUBW3	APK, P, 40(APK)		1824
			50	0080	CB	9A 00102		MOVZBL	176(SMQ), R0		1825
					50	D6 00107		INCL	R0		
	63	0080	CB		50	28 00109		MOV3	R0, 176(SMQ), (P)		
02	A8		53		58	A3 0010F	8\$:	SUBW3	APK, P, 2(APK)		1829
02	A7		53		57	A3 00114		SUBW3	ACR, P, 2(ACR)		1830
					04	00119		RET			1831

; Routine Size: 282 bytes, Routine Base: CODE + 0310

```
800 1832 1 ROUTINE RESOURCE_PACKET: L_RESOURCE_PACKET NOVALUE=
801 1833 1
802 1834 1 **
803 1835 1
804 1836 1 FUNCTIONAL DESCRIPTION:
805 1837 1 This routine builds the resource usage packet.
806 1838 1
807 1839 1 INPUT PARAMETERS:
808 1840 1 NONE
809 1841 1
810 1842 1 IMPLICIT INPUTS:
811 1843 1 ACM - Pointer to mailbox message.
812 1844 1 ACR - Pointer to accounting record buffer.
813 1845 1
814 1846 1 OUTPUT PARAMETERS:
815 1847 1 NONE
816 1848 1
817 1849 1 IMPLICIT OUTPUTS:
818 1850 1 Resource packet built in record buffer.
819 1851 1
820 1852 1 ROUTINE VALUE:
821 1853 1 NONE
822 1854 1
823 1855 1 SIDE EFFECTS:
824 1856 1 NONE
825 1857 1
826 1858 1 --
827 1859 1
828 1860 2 BEGIN
829 1861 2 EXTERNAL REGISTER
830 1862 2 ACM = ACCT_ACM_REG: REF BBLOCK, ! Pointer to mailbox message
831 1863 2 ACR = ACCT_ACR_REG: REF BBLOCK; ! Pointer to record buffer
832 1864 2 LOCAL
833 1865 2 APK: REF BBLOCK; ! Pointer to packet
834 1866 2 REGISTER
835 1867 2 P = 3; ! Pointer to free byte
836 1868 2
837 1869 2
838 1870 2 ACM_PACKET(ACR$K_RESOURCE, 0, .ACR, APK);
839 1871 2 MOV[3(
840 1872 2 %REF($BYTEOFFSET(ACM$K_VOLUMES) + 4 - $BYTEOFFSET(ACM$Q_LOGIN)),
841 1873 2 ACM[ACM$Q_LOGIN],
842 1874 2 APK[ACR$Q_LOGIN]; ... P);
843 1875 2 APK[ACR$W_LENGTH] = .P - .APK;
844 1876 2 ACR[ACR$W_LENGTH] = .P - .ACR;
845 1877 1 END;
```

013C 0000 RESOURCE_PACKET:

		58	02	A7	3C	00002	WORD	Save R2,R3,R4,R5,R8	
		58		57	C0	00006	MOVZWL	2(ACR), APK	1832
		68		8F	3C	00009	ADDL2	ACR, APK	1870
04	A8	44	A6	30	28	0000E	MOVZWL	#8197, (APK)	
							MOV[3	#48, 68(ACR), 4(APK)	1874

Accounting manager

15-SEP-1984 23:46:25
14-SEP-1984 12:36:55

VAX-11 BLISS-32 V4.0-742
[JOBCTL.SRC]ACCOUNTING.B32;1

Page 31
(9)

02 02 A8 A7

53

58	A3	00014
57	A3	00019
	04	0001E

SUBW3 APK, P, 2 (APK)
SUBW3 ACR, P, 2 (ACR)
RET

: 1875
: 1876
: 1877

; Routine Size: 31 bytes, Routine Base: CODE + 042A

ACC
V04

.....

```
847 1878 1 GLOBAL ROUTINE WRITE_USER_ACCOUNTING_RECORD(PACM,PSJH,PSMQ,LENGTH,ADDRESS): NOVALUE=
848 1879 1
849 1880 1 !++
850 1881 1
851 1882 1 FUNCTIONAL DESCRIPTION:
852 1883 1 This routine builds the user data record.
853 1884 1
854 1885 1 INPUT PARAMETERS:
855 1886 1 PACM - Pointer to mailbox message.
856 1887 1 PSJH - Pointer to SJH or 0.
857 1888 1 PSMQ - Pointer to SMQ or 0.
858 1889 1 LENGTH - Descriptor for user data.
859 1890 1 ADDRESS -
860 1891 1
861 1892 1 IMPLICIT INPUTS:
862 1893 1 NONE
863 1894 1
864 1895 1 OUTPUT PARAMETERS:
865 1896 1 NONE
866 1897 1
867 1898 1 IMPLICIT OUTPUTS:
868 1899 1 NONE
869 1900 1
870 1901 1 ROUTINE VALUE:
871 1902 1 NONE
872 1903 1
873 1904 1 SIDE EFFECTS:
874 1905 1 NONE
875 1906 1
876 1907 1 !--
877 1908 1
878 1909 2 BEGIN
879 1910 2 LOCAL
880 1911 2 ACR_BUFFER: BBLOCK[JB($K_MAXACCREC), ! Record buffer
881 1912 2 APK: REF BBLOCK; ! Pointer to packet
882 1913 2 REGISTER
883 1914 2 P = 3; ! Pointer to free byte
884 1915 2 GLOBAL REGISTER
885 1916 2 ACM = ACCT_ACM_REG: REF BBLOCK, ! Pointer to mailbox message
886 1917 2 ACR = ACCT_ACR_REG: REF BBLOCK, ! Pointer to record buffer
887 1918 2 SJH = ACCT_SJH_REG: REF BBLOCK, ! Pointer to SJH or 0
888 1919 2 SMQ = ACCT_SMQ_REG: REF BBLOCK; ! Pointer to SMQ or 0
889 1920 2
890 1921 2
891 1922 2 ACM = .PACM;
892 1923 2 ACR = ACR_BUFFER;
893 1924 2 SJH = .PSJH;
894 1925 2 SMQ = .PSMQ;
895 1926 2 ACM_RECORD(ACR$K_USER, 0, CUR_TIME, .ACR);
896 1927 2 IDENT_PACKET();
897 1928 2 ACM_PACKET(ACR$K_USER_DATA, 0, .ACR, APK);
898 1929 2 CH$DCHAR(.LENGTH, APK[ACR$K_USER_DATA]);
899 1930 2 MOV$3(LENGTH, .ADDRESS, APK[ACR$K_USER_DATA]+1; ... P);
900 1931 2 APK[ACR$K_LENGTH] = .P - .APK;
901 1932 2 ACR[ACR$K_LENGTH] = .P - .ACR;
902 1933 2 WRITE_ACCOUNTING_FILE();
903 1934 1 END;
```

				0AFC 00000				
			SE	FC00	CE	9E	00002	
			56	04	AC	D0	00007	
			57		6E	9E	0000B	
			59	08	AC	D0	0000E	
			58	0C	AC	D0	00012	
			67	000C2012	8F	D0	00016	
		04	A7	00000000	EF	7D	0001D	
		FE9D	CF		00	FB	00025	
			56	02	A7	3C	0002A	
			56		57	C0	0002E	
			66	200B	8F	3C	00031	
		04	A6	10	AC	90	00036	
05	A6	14	BC	10	AC	28	0003B	
02	A6		53		56	A3	00042	
02	A7		53		57	A3	00047	
		FB66	CF		00	FB	0004C	
					04	00	00051	

.ENTRY	WRITE_USER_ACCOUNTING_RECORD, Save R2,R3,-								1878
	R4,R5,R6,R7,R9,R11								
MOVAB	-1024(SP), SP								1922
MOVL	PACM, ACM								1923
MOVAB	ACR_BUFFER, ACR								1924
MOVL	PSJR, SJH								1925
MOVL	PSMQ, SMQ								1926
MOVL	#794642, (ACR)								
MOVQ	CUR_TIME, 4(ACR)								
CALLS	#0, IDENT_PACKET								1927
MOVZWL	2(ACR), APK								1928
ADDL2	ACR, APK								
MOVZWL	#8203, (APK)								
MOVB	LENGTH, 4(APK)								1929
MOVCL	LENGTH, @ADDRESS, 5(APK)								1930
SUBW3	APK, P, 2(APK)								1931
SUBW3	ACR, P, 2(ACR)								1932
CALLS	#0, WRITE_ACCOUNTING_FILE								1933
RET									1934

; Routine Size: 82 bytes, Routine Base: CODE + 0449

```
905 1935 1 ROUTINE WRITE_FILE_LINK_RECORD(TYPE,FAB): NOVALUE=
906 1936 1
907 1937 1 ++
908 1938 1
909 1939 1 FUNCTIONAL DESCRIPTION:
910 1940 1 This routine builds the accounting file forward and back link records.
911 1941 1
912 1942 1 INPUT PARAMETERS:
913 1943 1 TYPE - Record type (ACRSK_FILE_FL, ACRSK_FILE_BL).
914 1944 1 FAB - Pointer to FAB from which to obtain filespec.
915 1945 1
916 1946 1 IMPLICIT INPUTS:
917 1947 1 NONE
918 1948 1
919 1949 1 OUTPUT PARAMETERS:
920 1950 1 NONE
921 1951 1
922 1952 1 IMPLICIT OUTPUTS:
923 1953 1 NONE
924 1954 1
925 1955 1 ROUTINE VALUE:
926 1956 1 NONE
927 1957 1
928 1958 1 SIDE EFFECTS:
929 1959 1 Accounting record written.
930 1960 1
931 1961 1 --
932 1962 1
933 1963 2 BEGIN
934 1964 2 MAP
935 1965 2 FAB: REF BBLOCK; ! Pointer to FAB
936 1966 2 LOCAL
937 1967 2 ACR_BUFFER: BBLOCK[JBC$K_MAXACCREC], ! Record buffer
938 1968 2 APK: REF BBLOCK, ! Pointer to packet
939 1969 2 NAM: REF BBLOCK, ! Pointer to NAM block
940 1970 2 L: ! Length of filename
941 1971 2 REGISTER
942 1972 2 P = 3; ! Pointer to free byte
943 1973 2 GLOBAL REGISTER
944 1974 2 ACR = ACCT_ACR_REG: REF BBLOCK; ! Pointer to record buffer
945 1975 2
946 1976 2
947 1977 2 ACR = ACR_BUFFER;
948 1978 2 ACM_RECORD(.TYPE, 0, CUR TIME, .ACR);
949 1979 2 ACM_PACKET(ACRSK_FILENAME, 0, .ACR, APK);
950 1980 2 P = APK[ACR$T_FILENAME];
951 1981 2 NAM = .FAB[FAB$L_NAM];
952 1982 2 L = .NAM[NAM$B_RSL];
953 1983 2 (.P)<0,8> = .L;
954 1984 2 P = P + 1;
955 1985 2 MOV3(L, .NAM[NAM$B_RSA], .P; ... P);
956 1986 2 APK[ACR$W_LENGTH] = .P - .APK;
957 1987 2 ACR[ACR$W_LENGTH] = .P - .ACR;
958 1988 2 WRITE_ACCOUNTING_FILE();
959 1989 1 END;
```

				00FC 00000 WRITE_FILE LINK_RECORD:			
			5E	FC00	CE 9E 00002	.WORD Save R2,R3,R4,R5,R6,R7	1935
			57		6E 9E 00007	MOVAB -1024(SP), SP	
67	07		67	000C2000	8F D0 0000A	MOVAB ACR_BUFFER, ACR	1977
			01	04	AC F0 00011	MOVL #792624, (ACR)	1978
		04	A7	00000000	EF 7D 00017	INSV TYPE, #1, #7, (ACR)	
			56	02	A7 3C 0001F	MOVQ CUR TIME, 4(ACR)	
			56		57 C0 00023	MOVZWL 2(ACR), APK	1979
			66	2009	8F 3C 00026	ADDL2 ACR, APK	
			53	04	A6 9E 0002B	MOVZWL #8201, (APK)	
			50	08	AC D0 0002F	MOVAB 4(R6), P	1980
			50	28	A0 D0 00033	MOVL FAB, R0	1981
			51	03	A0 9A 00037	MOVL 40(R0), NAM	
			83		A0 9A 00037	MOVZBL 3(NAM), L	1982
			B0		51 90 0003B	MOV L, (P)+	1983
			53		51 28 0003E	MOVCL L, 24(NAM), (P)	1985
02	63	04	53		56 A3 00043	SUBW3 APK, P, 2(APK)	1986
02	A6		53		57 A3 00048	SUBW3 ACR, P, 2(ACR)	1987
	A7		CF		00 FB 0004D	CALLS #0, WRITE_ACCOUNTING_FILE	1988
		FB13			04 00052	RET	1989

; Routine Size: 83 bytes, Routine Base: CODE + 049B

```
961 1990 1 GLOBAL ROUTINE WRITE_ACCOUNTING_RECORD(SJH,SMQ,ACM,STS): NOVALUE=
962 1991 1
963 1992 1 ++
964 1993 1
965 1994 1 FUNCTIONAL DESCRIPTION:
966 1995 1 This routine builds and writes an accounting record for a process, a
967 1996 1 completed batch or symbiont job, or an incomplete job.
968 1997 1
969 1998 1 INPUT PARAMETERS:
970 1999 1 NONE
971 2000 1
972 2001 1 IMPLICIT INPUTS:
973 2002 1 SJH - Pointer to SJH or 0.
974 2003 1 SMQ - Pointer to SMQ or 0.
975 2004 1 ACM - Pointer to ACM or 0.
976 2005 1 STS - (Optional) Forced completion status.
977 2006 1
978 2007 1 OUTPUT PARAMETERS:
979 2008 1 NONE
980 2009 1
981 2010 1 IMPLICIT OUTPUTS:
982 2011 1 NONE
983 2012 1
984 2013 1 ROUTINE VALUE:
985 2014 1 NONE
986 2015 1
987 2016 1 SIDE EFFECTS:
988 2017 1 Accounting record written.
989 2018 1
990 2019 1 --
991 2020 1
992 2021 2 BEGIN
993 2022 2 MAP
994 2023 2 SJH: REF BBLOCK, ! Pointer to SJH
995 2024 2 SMQ: REF BBLOCK; ! Pointer to SMQ
996 2025 2 LOCAL
997 2026 2 LACM: BBLOCK[$BYTEOFFSET(ACMSW_IMAGENAME)+2]; ! Fake message
998 2027 2 BUILTIN
999 2028 2 ACTUALCOUNT;
1000 2029 2
1001 2030 2
1002 2031 2 IF .SMQ EQL 0
1003 2032 2 THEN
1004 2033 2 WRITE_PROCESS_RECORD(0, 0, .ACM)
1005 2034 2 ELSE
1006 2035 2 IF .SMQ[SMQSV_BATCH]
1007 2036 2 THEN
1008 2037 2 IF .ACM NEQ 0
1009 2038 2 THEN
1010 2039 2 WRITE_PROCESS_RECORD(.SJH, .SMQ, .ACM)
1011 2040 2 ELSE
1012 2041 2 BEGIN
1013 2042 2 CH$FILL(0, %ALLOCATION(LACM), LACM);
1014 2043 2 LACM[ACMSW_TYPE] = MSG$ DELPROC;
1015 2044 2 LACM[ACMSL_UIC] = .SJH[SJHSL_UIC];
1016 2045 2 CH$MOVE(SJR$S_USERNAME, SJH[SJH$T_USERNAME], LACM[ACM$T_USERNAME]);
1017 2046 2 CH$MOVE(SJH$S_ACCOUNT, SJH[SJH$T_ACCOUNT], LACM[ACM$T_ACCOUNT]);
```

```
1018 2047 3 BITVECTOR(LACM[ACMSL_STS], $BITPOSITION(PCBSV_BATCH)) = TRUE;
1019 2048 COPY TIME(CUR TIME, LACM[ACMSQ_LOGIN]);
1020 2049 IF ACTUALCOUNT() GTRU 3
1021 2050 THEN
1022 2051 BEGIN
1023 2052 LACM[ACMSL_FINALSTS] = .STS;
1024 2053 SJH[SJHSL_CONDITION_1] = .STS;
1025 2054 SJH[SJHSL_CONDITION_2] = 0;
1026 2055 SJH[SJHSL_CONDITION_3] = 0;
1027 2056 END;
1028 2057 WRITE_PROCESS_RECORD(.SJH, .SMQ, LACM);
1029 2058 END
1030 2059 ELSE
1031 2060 BEGIN
1032 2061 IF ACTUALCOUNT() GTRU 3
1033 2062 THEN
1034 2063 BEGIN
1035 2064 SJH[SJHSL_CONDITION_1] = .STS;
1036 2065 SJH[SJHSL_CONDITION_2] = 0;
1037 2066 SJH[SJHSL_CONDITION_3] = 0;
1038 2067 END;
1039 2068 IF .(SMQ[SMQSQ_ACM_BEGTIM]) EQL 0
1040 2069 THEN
1041 2070 COPY TIME(CUR TIME, SMQ[SMQSQ_ACM_BEGTIM]);
1042 2071 WRITE_PRINT_RECORD(.SJH, .SMQ);
1043 2072 END;
1044 2073 1 END;
```

				01FC 00000	.ENTRY	WRITE_ACCOUNTING_RECORD, Save R2,R3,R4,R5,-	1990
			58	00000000	EF 9E 00002	MOVAB R6,R7,R8	
			5E	84	AE 9E 00009	MOVAB -124(SP), SP	
			57	08	AC D0 0000D	MOVL SMQ, R7	2031
				0C	07 12 00011	BNEQ 1\$	
					AC DD 00013	PUSHL ACM	2033
					7E 7C 00016	CLRG -(SP)	
					52 11 00018	BRB 5\$	
			56	04	AC D0 0001A	MOVL SJH, R6	2039
			50	0C	A7 E9 0001E	BLBC 12(R7), 6\$	2035
				0C	AC D5 00022	TSTL ACM	2037
					05 13 00025	BEQL 2\$	
				0C	AC DD 00027	PUSHL ACM	2039
					3D 11 0002A	BRB 4\$	
007C	BF	00	6E	00 2C 0002C	2\$: MOVCS #0, (SP), #0, #124, LACM		2042
				6E	00033		
			6E	03 B0 00034	MOVW #3, LACM		2043
		0C	AE	0144 C6 D0 00037	MOVL 324(R6), LACM+12		2044
	10	AE	0148 C6	0C 28 0003D	MOVCS #12, 328(R6), LACM+16		2045
	1C	AE	14 A6	08 28 00044	MOVCS #8, 20(R6), LACM+28		2046
			2D	AE 40 8F 88 0004A	BISB2 #64, LACM+45		2047
			44	AE 68 7D 0004F	MOVQ CUR TIME, LACM+68		2048
				03	6C 91 00053	CMQB (APT, #3	2049
					0F 1B 00056	BLEQU 3\$	

ACCOUNTING
V04-000

Accounting manager

M 2
15-Sep-1984 23:46:25
14-Sep-1984 12:36:55

VAX-11 Bliss-32 V4.0-742
[JOBCTL.SRC]ACCOUNTING.B32;1

Page 38
(12)

4C	AE	10	AC	D0	00058	MOVL	STS, LACM+76	:	2052
00DC	C6	10	AC	D0	0005D	MOVL	STS, 220(R6)	:	2053
		00E0	C6	7C	00063	CLRQ	224(R6)	:	2054
			5E	DD	00067	PUSHL	SP	:	2057
0000V	7E		56	7D	00069	MOVQ	R6, -(SP)	:	
	CF		03	FB	0006C	CALLS	#3, WRITE_PROCESS_RECORD	:	
				04	00071	RET		:	2037
	03		6C	91	00072	CMPB	(AP), #3	:	2061
			0E	1B	00075	BLEQU	7\$:	
00DC	50	04	AC	D0	00077	MOVL	SJH, R0	:	2064
	C0	10	AC	D0	0007B	MOVL	STS, 220(R0)	:	
		00E0	C0	7C	00081	CLRQ	224(R0)	:	2065
		14	A7	D5	00085	TSTL	20(R7)	:	2068
			04	12	00088	BNEQ	8\$:	
14	A7		68	7D	0008A	MOVQ	CUR_TIME, 20(R7)	:	2070
	7E		56	7D	0008E	MOVQ	R6, -(SP)	:	2071
0000V	CF		02	FB	00091	CALLS	#2, WRITE_PRINT_RECORD	:	
			04	00096	RET			:	2073

; Routine Size: 151 bytes, Routine Base: CODE + 04EE

```
1046 2074 1 ROUTINE WRITE_PRINT_RECORD(PSJH,PSMQ): NOVALUE=
1047 2075 1
1048 2076 1 ++
1049 2077 1
1050 2078 1 FUNCTIONAL DESCRIPTION:
1051 2079 1 This routine builds the print accounting record.
1052 2080 1
1053 2081 1 INPUT PARAMETERS:
1054 2082 1 NONE
1055 2083 1
1056 2084 1 IMPLICIT INPUTS:
1057 2085 1 PSJH - Pointer to SJH.
1058 2086 1 PSMQ - Pointer to SMQ.
1059 2087 1
1060 2088 1 OUTPUT PARAMETERS:
1061 2089 1 NONE
1062 2090 1
1063 2091 1 IMPLICIT OUTPUTS:
1064 2092 1 NONE
1065 2093 1
1066 2094 1 ROUTINE VALUE:
1067 2095 1 NONE
1068 2096 1
1069 2097 1 SIDE EFFECTS:
1070 2098 1 Print accounting record written.
1071 2099 1
1072 2100 1 --
1073 2101 1
1074 2102 2 BEGIN
1075 2103 2 LOCAL
1076 2104 2 ACM_BUFFER: BBLOCK[$BYTEOFFSET(ACMSW_USERREQ)], ! Fake message
1077 2105 2 ACR_BUFFER: BBLOCK[JBC$K_MAXACCRC], ! Record buffer
1078 2106 2 APK: REF BBLOCK; ! Pointer to packet
1079 2107 2 GLOBAL REGISTER
1080 2108 2 ACM = ACCT_ACM_REG: REF BBLOCK, ! Pointer to accounting message
1081 2109 2 ACR = ACCT_ACR_REG: REF BBLOCK, ! Pointer to record buffer
1082 2110 2 SJH = ACCT_SJH_REG: REF BBLOCK, ! Pointer to SJH
1083 2111 2 SMQ = ACCT_SMQ_REG: REF BBLOCK, ! Pointer to SMQ
1084 2112 2
1085 2113 2
1086 2114 2 ACM = ACM_BUFFER;
1087 2115 2 SJH = .PSJH;
1088 2116 2 SMQ = .PSMQ;
1089 2117 2 CH$FILL(0, %ALLOCATION(ACM_BUFFER), .ACM);
1090 2118 2 ACM[ACMSL_PID] = .SJH[SJHSL_PID];
1091 2119 2 ACM[ACMSL_UIC] = .SJH[SJHSL_UIC];
1092 2120 2 ACM[ACMSB_PROCPRI] = .SJH[SJH$B_PRIORITY];
1093 2121 2 CH$MOVE(ACM$S_USERNAME, SJH[SJH$T_USERNAME], ACM[ACM$T_USERNAME]);
1094 2122 2 CH$MOVE(ACM$S_ACCOUNT, SJH[SJH$T_ACCOUNT], ACM[ACM$T_ACCOUNT]);
1095 2123 2
1096 2124 2
1097 2125 2 ACR = ACR_BUFFER;
1098 2126 2 ACM_RECORD(ACR$K_PRINT, 0, CUR_TIME, .ACR);
1099 2127 2 IDENT_PACKET();
1100 2128 2 ACM_PACKET(ACR$K_PRINT, 0, .ACR, APK);
1101 2129 2 APK[ACR$K_PRINTSTS] = .SJH[SJH$K_CONDITION 1];
1102 2130 2 COPY_TIME[SJH$K_TIME], APK[ACR$K_QUE_TIME]);
```

```

1103 2131 2 COPY TIME(SMQ[SMQSQ_ACM_BEGTIM], APK[ACRSQ_BEGTIM]);
1104 2132 2 APK[ACRSL_SYMCPUTIM] = .SMQ[SMQSL_ACM_SYMCPUTIM];
1105 2133 2 APK[ACRSL_PAGECNT] = .SMQ[SMQSL_ACM_PAGECNT];
1106 2134 2 APK[ACRSL_QIOCNT] = .SMQ[SMQSL_ACM_QIOCNT];
1107 2135 2 APK[ACRSL_GETCNT] = .SMQ[SMQSL_ACM_GETCNT];
1108 2136 2 APK[ACRSW_LENGTH] = $BYTEOFFSET(ACRSL_GETCNT) + 4;
1109 2137 2 ACR[ACRSW_LENGTH] = .APK + $BYTEOFFSET(ACRSL_GETCNT) + 4 - .ACR;
1110 2138 2 WRITE_ACCOUNTING_FILE();
1111 2139 2
1112 2140 2
1113 2141 2 CLEAR TIME(SMQ[SMQSQ_ACM_BEGTIM]);
1114 2142 2 SMQ[SMQSL_ACM_SYMCPUTIM] = 0;
1115 2143 2 SMQ[SMQSL_ACM_PAGECNT] = 0;
1116 2144 2 SMQ[SMQSL_ACM_QIOCNT] = 0;
1117 2145 2 SMQ[SMQSL_ACM_GETCNT] = 0;
1118 2146 1 END;

```

DAFC 00000 WRITE_PRINT_RECORD:

ADDRESS		HEX	ASCII	OPERATION	COMMENT	PC
0044	8F	00		MOVAB	Save R2,R3,R4,R5,R6,R7,R9,R11	2074
				MOVAB	-1092(SP), SP	2114
				MOVAB	ACM BUFFER, ACM	2115
				MOVAB	PSJH, SJH	2116
				MOVAB	PSMQ, SMQ	2117
				MOVAB	#0, (SP), #0, #68, (ACM)	2118
				MOVAB	304(SJH), 40(ACM)	2119
				MOVAB	324(SJH), 12(ACM)	2120
				MOVAB	381(SJH), 36(ACM)	2121
				MOVAB	#12, 328(SJH), 16(ACM)	2122
				MOVAB	#8, 20(SJH), 28(ACM)	2125
				MOVAB	ACR BUFFER, ACR	2126
				MOVAB	#79, 640, (ACR)	2127
				MOVAB	CUR TIME, 4(ACR)	2128
				CALLS	#0, IDENT PACKET	2129
				MOVZWL	2(ACR), APK	2130
				ADDL2	ACR, APK	2131
				MOVZWL	#8209, (APK)	2132
				MOVAB	220(SJH), 4(APK)	2133
				MOVAB	316(SJH), 8(APK)	2134
				MOVAB	20(SMQ), 16(APK)	2135
				MOVAB	40(SMQ), 24(APK)	2136
				MOVAB	32(SMQ), 28(APK)	2137
				MOVAB	28(SMQ), 36(APK)	2138
				MOVW	#40, 2(APK)	2139
				SUBL2	ACR, R0	2140
				ADDW3	#40, R0, 2(ACR)	2141
				CALLS	#0, WRITE_ACCOUNTING_FILE	2142
				CLRQ	20(SMQ)	2143
				CLRQ	28(SMQ)	2144
				CLRQ	36(SMQ)	2145
				RET		2146

; Routine Size: 152 bytes, Routine Base: CODE + 0585

ACCOUNTING
V04-000

Accounting manager

C 3
15-Sep-1984 23:46:25
14-Sep-1984 12:36:55

VAX-11 Bliss-32 V4.0-742
[JOBCTL.SRC]ACCOUNTING.B32;1

Page 41
(13)

AS
V04

```
1120 2147 1 ROUTINE WRITE_PROCESS_RECORD(P SJH,PSMQ,PACM): NOVALUE=
1121 2148
1122 2149 1 ++
1123 2150
1124 2151 1 FUNCTIONAL DESCRIPTION:
1125 2152 1 This routine builds process deletion, process purge, image deletion,
1126 2153 1 image purge, login failure, and system initialization records.
1127 2154 1
1128 2155 1 INPUT PARAMETERS:
1129 2156 1 NONE
1130 2157 1
1131 2158 1 IMPLICIT INPUTS:
1132 2159 1 PSJH - Pointer to SJH or 0.
1133 2160 1 PSMQ - Pointer to SMQ or 0.
1134 2161 1 PACM - Pointer to mailbox message.
1135 2162 1
1136 2163 1 OUTPUT PARAMETERS:
1137 2164 1 NONE
1138 2165 1
1139 2166 1 IMPLICIT OUTPUTS:
1140 2167 1 NONE
1141 2168 1
1142 2169 1 ROUTINE VALUE:
1143 2170 1 NONE
1144 2171 1
1145 2172 1 SIDE EFFECTS:
1146 2173 1 NONE
1147 2174 1
1148 2175 1 --
1149 2176 1
1150 2177 2 BEGIN
1151 2178 2 LOCAL
1152 2179 2 ACR_BUFFER: BBLOCK[EJBC$K_MAXACCREC], ! Record buffer
1153 2180 2 TYPE, ! Record type
1154 2181 2 SUBTYPE; ! Record subtype
1155 2182 2 GLOBAL REGISTER
1156 2183 2 ACM = ACCT_ACM_REG: REF BBLOCK, ! Pointer to mailbox message
1157 2184 2 ACR = ACCT_ACR_REG: REF BBLOCK, ! Pointer to record buffer
1158 2185 2 SJH = ACCT_SJH_REG: REF BBLOCK, ! Pointer to SJH or 0
1159 2186 2 SMQ = ACCT_SMQ_REG: REF BBLOCK; ! Pointer to SMQ or 0
1160 2187 2
1161 2188 2
1162 2189 2 ACM = .PACM;
1163 2190 2 ACR = ACR_BUFFER;
1164 2191 2 SJH = .PSJH;
1165 2192 2 SMQ = .PSMQ;
1166 2193 2 IF .SJH NEQ 0 THEN ACM[ACM$S_FINALSTS] = .SJH[SJH$S_CONDITION_1];
1167 2194 2 SUBTYPE = 0;
1168 2195 2 IF
1169 2196 2 BEGIN
1170 2197 2 CASE .ACM[ACM$W_TYPE] FROM MSG$_DELPROC TO MSG$_PURIMAG OF
1171 2198 2 SET
1172 2199 2
1173 2200 2 [INRANGE, OUTFRANGE]:
1174 2201 2 RETURN;
1175 2202 2
1176 2203 2 !+
```

```
1177 2204 DELPROC messages generate one of the following accounting records:
1178 2205
1179 2206     SYSINIT   System Initialization; the STARTUP process terminated
1180 2207     LOGFAIL   Login Failure; LOGINOUT terminated with an authorization error
1181 2208     PRCDL    Normal process termination
1182 2209
1183 2210 Account names starting with a binary null are special, reserved to
1184 2211 Digital, account names. These special account names are used to determine
1185 2212 what kind to accounting record to generate here.
1186 2213
1187 2214 The system STARTUP process starts out life with an account name of all
1188 2215 binary nulls. This is changed to a single binary null followed by
1189 2216 '<start>' when it gets set logged in by LOGINOUT. Furthermore, the
1190 2217 STARTUP process has no terminal and a username of SYSTEM. These
1191 2218 characteristics are checked and, if met, cause a SYSINIT record.
1192 2219
1193 2220 A login failure has an account name starting with a single binary
1194 2221 null followed by some descriptive keyword enclosed by <>'s. This
1195 2222 special account name is set up by LOGINOUT whenever it must terminate
1196 2223 due to an authorization failure. Account names of this special
1197 2224 form cause a LOGFAIL record. The following special account name
1198 2225 descriptive keywords are currently used:
1199 2226
1200 2227     <batch>   Batch job login failure
1201 2228     <det>     Detached process login failure
1202 2229     <login>   Interactive login failure
1203 2230     <net>     Network login failure
1204 2231
1205 2232 Otherwise, a PRCDL record is generated.
1206 2233
1207 2234 [MSG$ DELPROC]:
1208 2235     BEGIN
1209 2236     MACRO
1210 2237         SYSINIT_ACCOUNT = %STRING(%CHAR(0), '<start>')%;
1211 2238     LOCAL
1212 2239         NON_NULLS;
1213 2240     SKPC(%REF(0), %REF(ACM$S_ACCOUNT), ACM[ACM$T_ACCOUNT]; NON_NULLS);
1214 2241     IF (.NON_NULLS EQL 0
1215 2242         OR
1216 2243         CH$EQL(
1217 2244             ACM$S_ACCOUNT, ACM[ACM$T_ACCOUNT],
1218 2245             %CHARCOUNT(SYSINIT_ACCOUNT), UPLIT BYTE(SYSINIT_ACCOUNT),
1219 2246             %C' ')
1220 2247         AND CH$RCHAR(ACM[ACM$T_TERMINAL]) EQL 0
1221 2248         AND CH$EQL(
1222 2249             ACM$S_USERNAME, ACM[ACM$T_USERNAME],
1223 2250             %CHARCOUNT('SYSTEM'), UPLIT BYTE('SYSTEM'),
1224 2251             %C' ')
1225 2252         THEN
1226 2253         BEGIN
1227 2254             ACM RECORD(ACR$K_SYSINIT, 0, CUR_TIME, .ACR);
1228 2255             IDENT PACKET();
1229 2256             RESOURCE PACKET();
1230 2257             WRITE ACCOUNTING_FILE();
1231 2258             RETURN;
1232 2259         END;
1233 2260
```

```
.. 1234      2261      5      IF (.NON_NULLS EQL 0
.. 1235      2262      5      AND
.. 1236      2263      5      CH$RCHAR(ACM[ACM$T_TERMINAL]) NEQ 0)
.. 1237      2264      5      OR .NON_NULLS EQL ACM$$_ACCOUNT - 1
.. 1238      2265      5      THEN
.. 1239      2266      5      BEGIN
.. 1240      2267      5      TYPE = ACR$K_LOGFAIL;
.. 1241      2268      5      FALSE
.. 1242      2269      5      END
.. 1243      2270      5      ELSE
.. 1244      2271      5      BEGIN
.. 1245      2272      5      TYPE = ACR$K_PRCDEL;
.. 1246      2273      5      TRUE
.. 1247      2274      5      END
.. 1248      2275      5      END;
.. 1249      2276      5
.. 1250      2277      5      [MSG$ PURPROC]:
.. 1251      2278      5      BEGIN
.. 1252      2279      5      TYPE = ACR$K_PRCPUR;
.. 1253      2280      5      TRUE
.. 1254      2281      5      END;
.. 1255      2282      5
.. 1256      2283      5      [MSG$ DELIMAG]:
.. 1257      2284      5      BEGIN
.. 1258      2285      5      TYPE = ACR$K_IMGDEL;
.. 1259      2286      5      TRUE
.. 1260      2287      5      END;
.. 1261      2288      5
.. 1262      2289      5      [MSG$ PURIMAG]:
.. 1263      2290      5      BEGIN
.. 1264      2291      5      TYPE = ACR$K_IMGPUR;
.. 1265      2292      5      TRUE
.. 1266      2293      5      END;
.. 1267      2294      5
.. 1268      2295      5      TES
.. 1269      2296      5      END
.. 1270      2297      5      THEN
.. 1271      2298      5      BEGIN
.. 1272      2299      5      IF CH$RCHAR(ACM[ACM$T_TERMINAL]) NEQ 0
.. 1273      2300      5      THEN
.. 1274      2301      5      SUBTYPE = ACR$K_INTERACTIVE
.. 1275      2302      5
.. 1276      2303      5      ELSE IF .BITVECTOR[ACM[ACM$$_STS], $BITPOSITION(PCB$V_BATCH)]
.. 1277      2304      5      THEN
.. 1278      2305      5      SUBTYPE = ACR$K_BATCH
.. 1279      2306      5
.. 1280      2307      5      ELSE IF .BITVECTOR[ACM[ACM$$_STS], $BITPOSITION(PCB$V_NETWRK)]
.. 1281      2308      5      THEN
.. 1282      2309      5      SUBTYPE = ACR$K_NETWORK
.. 1283      2310      5
.. 1284      2311      5      ELSE IF .ACM[ACM$$_OWNER] NEQ 0
.. 1285      2312      5      THEN
.. 1286      2313      5      SUBTYPE = ACR$K_SUBPROCESS
.. 1287      2314      5
.. 1288      2315      5      ELSE
.. 1289      2316      5      SUBTYPE = ACR$K_DETACHED;
.. 1290      2317      5      END;
```

```

1291 2318 22
1292 2319 22
1293 2320 22 ACM_RECORD(.TYPE, .SUBTYPE, CUR_TIME, .ACR);
1294 2321 22 IDENT_PACKET();
1295 2322 22 RESOURCE_PACKET();
1296 2323 22
1297 2324 22
1298 2325 22 IF ONEOF_(.TYPE, BMSK_(ACRSK_IMGPUR, ACRSK_IMGDEL))
1299 2326 22 THEN
1300 2327 22     BEGIN
1301 2328 22     LOCAL
1302 2329 22         APK:                REF BBLOCK,                ! Pointer to packet
1303 2330 22         Q:                REF VECTOR[.BYTE];            ! Pointer to image name string
1304 2331 22     REGISTER
1305 2332 22         P = 3;                ! Pointer to free byte
1306 2333 22
1307 2334 22
1308 2335 22     ACM_PACKET(ACRSK_IMAGENAME, 0, .ACR, APK);
1309 2336 22     Q = .ACM + .ACM[ACMSW_IMAGENAME];
1310 2337 22     MOV(3(XREF(.Q[0] + 1), .Q, APK[ACRST_IMAGENAME]; ... P);
1311 2338 22     APK[ACRSW_LENGTH] = .P - .APK;
1312 2339 22     ACR[ACRSW_LENGTH] = .P - .ACR;
1313 2340 22     END;
1314 2341 22
1315 2342 22
1316 2343 22 WRITE_ACCOUNTING_FILE();
1317 2344 22 END;

```

```

3E 74 72 61 74 73 3C 00 0061D P.AAC: .ASCII <0>\<start>\
      4D 45 54 53 59 53 00625 P.AAD: .ASCII \SYSTEM\

```

[illegible]

[illegible]

ACCOUNTING
V04-000

Accounting manager

1 3
15-Sep-1984 23:46:25
14-Sep-1984 12:36:55

VAX-11 Bliss-32 V4.0-742
[JOBCTL.SRC]ACCOUNTING.B32;1

Page 47
(14)

50 18000000

8F

52 78 000F6

ASHL TYPE, #402653184, R0

: 2325

59

02

27 18 000FE

BGEQ 19\$

: 2335

59

2007

A7 3C 00100

MOVZWL 2(ACR), APK

69

7A

57 C0 00104

ADDL2 ACR, APK

51

8F 3C 00107

MOVZWL #8199, (APK)

51

A6 3C 0010C

MOVZWL 122(ACM), Q

: 2336

50

56 C0 00110

ADDL2 ACM, Q

: 2337

04 A9

61

61 9A 00113

MOVZBL (Q), R0

02 A9

53

50 D6 00116

INCL R0

02 A7

53

59 A3 0011D

MOVCL R0, (Q), 4(APK)

: 2338

FCFO

53

57 A3 00122

SUBW3 APK, P, 2(APK)

: 2339

00 FB 00127

SUBW3 ACR, P, 2(ACR)

: 2343

19\$:

CALLS #0, WRITE_ACCOUNTING_FILE

: 2344

20\$:

RET

; Routine Size: 301 bytes, Routine Base: CODE + 062B

```
1319 2345 1 GLOBAL ROUTINE PROCESS_ACCOUNTING: NOVALUE=
1320 2346 1
1321 2347 1 ++
1322 2348 1
1323 2349 1 FUNCTIONAL DESCRIPTION:
1324 2350 1 This routine processes the message types:
1325 2351 1 MSGS_PURPROC process purge
1326 2352 1 MSGS_DELMAG image deletion
1327 2353 1 MSGS_PURIMAG image purge
1328 2354 1 by writing an accounting record.
1329 2355 1
1330 2356 1 INPUT PARAMETERS:
1331 2357 1 NONE
1332 2358 1
1333 2359 1 IMPLICIT INPUTS:
1334 2360 1 MBX - Pointer to buffered mailbox message.
1335 2361 1
1336 2362 1 OUTPUT PARAMETERS:
1337 2363 1 NONE
1338 2364 1
1339 2365 1 IMPLICIT OUTPUTS:
1340 2366 1 NONE
1341 2367 1
1342 2368 1 ROUTINE VALUE:
1343 2369 1 NONE
1344 2370 1
1345 2371 1 SIDE EFFECTS:
1346 2372 1 Accounting record written.
1347 2373 1
1348 2374 1 --
1349 2375 1
1350 2376 2 BEGIN
1351 2377 2 LOCAL
1352 2378 2 SJH_N, ! Record number of SJH
1353 2379 2 SMQ_N; ! Record number of SMQ
1354 2380 2
1355 2381 2
1356 2382 2 IF
1357 2383 2 BEGIN
1358 2384 2 IF .BITVECTOR[MBX[ACMSL_STS], $BITPOSITION(PCBSV_BATCH)]
1359 2385 2 THEN
1360 2386 2 FIND_PROCESS_DATA(
1361 2387 2 PDE_K_BATCH, MBX[ACMSL_PID], FALSE;
1362 2388 2 , SMQ_N, SJH_N)
1363 2389 2 ELSE
1364 2390 2 FALSE
1365 2391 2 END
1366 2392 2 THEN
1367 2393 2 BEGIN
1368 2394 2 LOCK_QUEUE_FILE();
1369 2395 2 WRITE_PROCESS_RECORD(
1370 2396 2 READ_RECORD(.SJH_N),
1371 2397 2 READ_RECORD(.SMQ_N),
1372 2398 2 MBX);
1373 2399 2 UNLOCK_QUEUE_FILE();
1374 2400 2 END
1375 2401 2 ELSE
```

; 1376
; 13772402 2 WRITE_PROCESS_RECORD(0, 0, .MBX);
2403 1 END;

			DEOC 00000		.ENTRY	PROCESS ACCOUNTING, Save R2,R3,R9,R10,R11	: 2345
	53	00000000G	EF 9E 00002		MOVAB	READ_RECORD, R3	
	52	00000000'	EF 9E 00009		MOVAB	MBX, R2	
	50		62 D0 00010		MOVL	MBX, R0	: 2384
35	2D	A0	06 E1 00013		BBC	#6, 45(R0), 1\$	
			7E D4 00018		CLRL	-(SP)	: 2386
			A0 DD 0001A		PUSHL	40(R0)	: 2387
		28	01 DD 0001D		PUSHL	#1	: 2386
	00000000G	EF	03 FB 0001F		CALLS	#3, FIND_PROCESS_DATA	
	24		50 E9 00026		BLBC	R0, 1\$	
	00000000G	EF	00 FB 00029		CALLS	#0, LOCK_QUEUE_FILE	: 2394
			62 DD 00030		PUSHL	MBX	: 2398
			5A DD 00032		PUSHL	SMQ_N	: 2397
	63		01 FB 00034		CALLS	#1, READ_RECORD	
			50 DD 00037		PUSHL	R0	
			5B DD 00039		PUSHL	SJH_N	: 2396
	63		01 FB 0003B		CALLS	#1, READ_RECORD	
			50 DD 0003E		PUSHL	R0	
	FE8E	CF	03 FB 00040		CALLS	#3, WRITE_PROCESS_RECORD	
	00000000G	EF	00 FB 00045		CALLS	#0, UNLOCK_QUEUE_FILE	: 2399
			04 0004C		RET		: 2382
			62 DD 0004D 1\$:		PUSHL	MBX	: 2402
			7E 7C 0004F		CLRL	-(SP)	
	FE7D	CF	03 FB 00051		CALLS	#3, WRITE_PROCESS_RECORD	
			04 00056		RET		: 2403

; Routine Size: 87 bytes, Routine Base: CODE + 0758

ACCOUNTNG
V04-000

Accounting manager

L 3
15-Sep-1984 23:46:25
14-Sep-1984 12:36:55

VAX-11 Bliss-32 V4.0-742
[JOBCTL.SRC]ACCOUNTNG.B32;1

Page 50
(16)

: 1379
: 1380

2404 1 END
2405 0 ELUDOM

.EXTRN LIB\$SIGNAL

PSECT SUMMARY

Name	Bytes	Attributes
COMMON	5024	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, OVR, NOPIC, ALIGN(2)
CODE	1967	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics

File	Symbols		Pages Mapped	Processing Time
	Total	Loaded Percent		
_\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	220 1	1000	00:01.5

COMMAND QUALIFIERS

: BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LISS:ACCOUNTNG/OBJ=OBJ\$:ACCOUNTNG MSRC\$:ACCOUNTNG/UPDATE=(ENH\$:ACCOUNTNG)

: Size: 1928 code + 5063 data bytes
: Run Time: 00:39.4
: Elapsed Time: 02:39.7
: Lines/CPU Min: 3661
: Lexemes/CPU-Min: 46131
: Memory Used: 392 pages
: Compilation Complete

0190

AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

0191 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

BATCH
LIS

BROADCAST
LIS

BUFFERS
LIS

CONTROL
LIS

ASYNCHRON
LIS

CHECKPROT
LIS